UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

ANALÝZA ČASOVÝCH PÁSIEM

DIPLOMOVÁ PRÁCA

2026 **Matej DUKÁT**

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY

ANALÝZA ČASOVÝCH PÁSIEM

DIPLOMOVÁ PRÁCA

Študijný program:	Ekonomická a finančná matematika
Študijný odbor:	1114 Aplikovaná matematika
Školiace pracovisko:	Katedra aplikovanej matematiky a štatistiky
Vedúci práce:	Mgr. Juraj Hledík, PhD.

Bratislava 2026 **Matej DUKÁT**





Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Matej Dukát
Študijný program: ekonomicko-finančná matematika a modelovanie (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: matematika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Analýza časových pásiem
A time zone analysis

Anotácia: Geografickú polohu užívateľov siete Bitcoin je kvôli inherentnej pseudoanonymite jej protokolu zložité určiť. Táto práca má za úlohu vyvinúť algoritmus na aproximáciu zemepisnej dĺžky pre polohu používateľov siete Bitcoin na základe časov transakcií zaznamenaných na BTC blockchain. Práca si vyžaduje vynikajúcu znalosť programovania a práce s veľkými dátami.

Vedúci: Mgr. Juraj Hledík, PhD.
Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: doc. Mgr. Igor Melicherčík, PhD.

Dátum zadania: 09.01.2025

Dátum schválenia: 15.01.2025

prof. RNDr. Daniel Ševčovič, DrSc.
garant študijného programu

.....
študent

.....
vedúci práce

Čestné vyhlásenie

Čestne vyhlasujem, že celú diplomovú prácu na tému „Analýza časových pásiem“, vrátane všetkých jej príloh a obrázkov, som vypracoval samostatne, a to s použitím literatúry uvedenej v priloženom zozname a nástrojov umelej inteligencie. Vyhlasujem, že nástroje umelej inteligencie som použil v súlade s príslušnými právnymi predpismi, akademickými právami a slobodami, etickými a morálnymi zásadami za súčasného dodržania akademickej integrity a že ich použitie je v práci vhodným spôsobom označené.

Miesto a dátum: ..... Titul, meno a priezvisko: .....

Abstrakt v štátnom jazyku

DUKÁT, Matej: Analýza časových pásiem [Diplomová práca], Univerzita Komenského v Bratislave, Fakulta matematiky, fyziky a informatiky, Katedra aplikovanej matematiky a štatistiky, školiteľ: Mgr. Juraj Hledík, PhD, Bratislava, 2026, 88s.

Predkladaná diplomová práca sa venuje návrhu klasifikačného modelu na predikciu geografickej polohy a časového pásma používateľov siete Bitcoin na základe ich transakčnej aktivity zaznamenanej na blockchaine. Práca sa zaoberá tvorbou unikátnej dátovej sady, čo zahŕňa extrakciu adries z fóra Bitcointalk, filtráciu komerčných entít a rozšírenie transakčnej histórie pomocou heuristického zhlukovania. Na dáta agregované do štyroch geografických regiónov sú následne aplikované modely logistickej regresie a XGBoostu. Výsledky ukazujú, že transakčná aktivita umožňuje identifikovať okno neaktivity používateľa, ktoré umožňuje rozlíšinie jednotlivých regiónov. Avšak už pri agregovaných regiónoch naráža model na limity rozlíšenia blízkych časových zón. V predkladanom diele je tiež prezentovaná experimentálna aplikácia metód semi-supervised learningu, ktorá však v tomto špecifickom kontexte nepreukázala zlepšenie predikčnej výkonnosti.

Kľúčové slová: Bitcoin, analýza časových pásiem, strojové učenie, klasifikačné modely.

Abstract

DUKÁT, Matej: A Time zone analysis [Master's Thesis], Comenius University Bratislava, Faculty of Mathematics, Physics and Informatics, Department of Applied Mathematics and Statistics, supervisor: Mgr. Juraj Hledík, PhD, Bratislava, 2026, 88 p.

The presented master's thesis focuses on the design of a classification model for predicting the geographical location and time zone of Bitcoin network users based on their transaction activity recorded on the blockchain. The thesis deals with the creation of a unique dataset, which involves extracting addresses from the Bitcointalk forum, filtering out commercial entities, and expanding the transaction history using heuristic clustering. Logistic regression and XGBoost models are subsequently applied to the data aggregated into four geographical regions. The results show that transaction activity allows for the identification of a user's window of inactivity, which enables the differentiation of individual regions. However, even with aggregated regions, the model encounters limits in distinguishing adjacent time zones. The presented work also includes an experimental application of semi-supervised learning methods, which, however, did not demonstrate an improvement in predictive performance in this specific context.

Keywords: Bitcoin, time zone analysis, machine learning, classification models.

Obsah

Úvod **7**

1 Základné koncepty a princíp fungovania kryptomeny Bitcoin **9**

1.1 Kryptografia a kryptomeny 9

1.1.1 Hašovacia funkcia 10

1.1.2 Digitálny podpis 11

1.2 Bitcoin a jeho architektúra 12

1.2.1 Klasifikácia a mechanizmus bitcoinových peňaženiek 13

1.2.2 Metódy nadobudnutia bitcoinu 16

1.2.3 Mechanizmus realizácie a propagácie transakcií 17

1.2.4 Memory pool 24

1.2.5 Schvaľovanie transakcií - Ťaženie 25

1.2.6 Burzy a ich architektúra 27

2 Dáta **31**

2.1 Krok 1 - bitcoinové adresy s polohou 31

2.2 Krok 2 - validácia adries a zber dát o transakciách 33

2.3 Krok 3 - priraďovanie časových zón a validácia lokality 35

2.4 Krok 4 - vylúčenie adries patriacich burzám 35

2.5 Krok 5 - heuristické zhlukovanie adries 36

2.6 Krok 6 - transakcie a regionálne zhlukovanie časových pásiem 39

3 Klasifikačné modely **43**

3.1 Logistická regresia 44

3.2 XGBoost 47

3.2.1 Rozhodovacie stromy 47

3.2.2 Boosting 51

3.2.3 Gradient boosting 52

3.2.4 XGBoost 56

3.3 Kritériá kvality klasifikačných modelov 59

4 Aplikácia klasifikačných modelov na dáta **62**

4.1 Analýza výsledkov použitia logistickej regresie 62

4.2 Hľadanie optimálnych parametrov XGBoostu 66

4.3 Analýza výsledkov použitia XGBoostu 69

5 Semi-supervised learning **73**

5.1 Samoučenie (Self-training) 74

5.2 Šírenie označení (Label Propagation) 75

Záver **79**

Zoznam použitej literatúry **82**

Databáza a kódy **84**

Príloha A **87**

Príloha B **88**

Úvod

Využívanie kryptomien, predovšetkým Bitcoinu, je v dnešnej dobe aktuálna a populárna téma. Ľudia za vidinou decentralizácie, slobody nakladania s vlastným majetkom a anonymity vyhľadávajú často túto digitálnu sieť. Najmä absencia centrálnej autority a otvorenosť systému spôsobili nárast popularity a zvýšený dopyt po participácii v takomto finančnom ekosystéme. Tento dopyt spôsobil nárast ceny a popularity kryptomien medzi investormi, čím sa kryptomeny čiastočne pretransformovali na investičné aktívum.

Dobrou otázkou však zostáva, odkiaľ títo používatelia pochádzajú? V tejto práci sa pozrieme najmä na to, ako možno matematicky a dátovo analyzovať správanie používateľov v sieti Bitcoin a na základe ich transakčnej aktivity predikovať ich geografický región, respektíve časové pásmo.

Základný koncept siete Bitcoin sformuloval v roku 2008 Satoshi Nakamoto, ktorého práca [14] obsahuje predstavenie kľúčových kryptografických techník používaných pri Bitcoine. Jedným z hlavných princípov Nakamotovej práce bola verejná účtovná kniha (blockchain), do ktorej sa trvalo a nemenne zapisujú všetky transakcie. Zatiaľ čo dáta o transakciách sú plne transparentné, samotná sieť je pseudonymná. Používatelia nevystupujú pod svojimi reálnymi identitami, ale sú reprezentovaní kryptografickými adresami.

Takáto anonymizácia výrazne komplikuje možnosti analýzy a získania relevantných dát s priradenými označeniami geografickej polohy používateľov. Existujú však rôzne alternatívne zdroje, pomocou ktorých je možné používateľov čiastočne deanonymizovať a získať relevantné informácie pre potreby analýz.

V tejto práci od základov vysvetlíme proces získavania dát z blockchainu a ich prepojenie s reálnymi používateľmi. Okrem predstavenia teórie fungovania Bitcoinu sa zameriame na využitie metód strojového učenia pri analýze denných cyklov aktivity používateľov. Naším cieľom bolo zistiť, či je možné na základe času vykonávania transakcií a identifikácie okna neaktivity úspešne predikovať, z akého časového pásma, respektíve globálneho regiónu daná entita pochádza. Takisto sme chceli preskúmať, ako sa s týmto problémom vysporiadajú modely logistickej regresii a XGBoostu. Nakoniec sme otestovali použitie metód semi-supervised learningu a

ich vplyv na výkonnosť modelu.

Práca je rozdelená na päť kapitol. V prvej kapitole uvedieme čitateľa do problematiky a princípov fungovania kryptomeny Bitcoin. Predstavíme architektúru transakcií, generovanie adries, fungovanie peňaženiek a búrz, čo je kľúčové pre pochopenie toho, ako možno deanonymizovať používateľov.

Druhá kapitola predstavuje proces získavania a prípravy dát. Popisujeme v nej extrakciu bitcoinových adries a polôh používateľov z fóra Bitcointalk, zber transakčných dát prostredníctvom API rozhrania a proces heuristického zhlukovania adries do entít. Záverom kapitoly definujeme rozdelenie na štyri základné geografické regióny.

V tretej kapitole predstavujeme matematické pozadie klasifikačných modelov, ktoré slúžia na samotnú predikciu regiónov. Zameriavame sa primárne na teóriu logistickej regresie a stromového algoritmu XGBoost. Takisto v nej definujeme metriky a hodnotiace kritériá, pomocou ktorých budeme analyzovať predikčnú schopnosť týchto klasifikátorov.

Štvrtá kapitola slúži na prezentáciu a vyhodnotenie výsledkov. Predstavujeme v nej výsledky natrénovaných modelov z predchádzajúcej kapitoly, aplikované na reálne zozbierané dáta. Rozoberieme si agregované výsledky prostredníctvom klasifikačných reportov a kalibračných grafov, ale pozrieme sa aj na detailnú manuálnu analýzu konkrétnych chýb modelu a ich možných príčin.

V piatej kapitole rozširujeme našu analytickú časť o metódy semi-supervised learningu. Tieto modely, konkrétne Self-training a Label Propagation, aplikujeme na rozsiahlu vzorku neoznačených dát, čím simulujeme reálne prostredie blockchainu, kde u drvivej väčšiny subjektov nepoznáme ich skutočnú polohu.

Vlastný prínos práce spočíva vo vytvorení unikátneho datasetu, ktorý spája údaje získané web scrapingom so surovými on-chain dátami a metódami heuristického zhlukovania. Na túto dátovú sadu sme následne aplikovali algoritmy strojového učenia, čím sme vytvorili klasifikačný model, ktorý odhaľuje limity geografickej klasifikácie.

1 Základné koncepty a princíp fungovania kryptomeny Bitcoin

Úvodná kapitola tejto práce slúži ako teoretický a technický základ pre pochopenie fungovania siete Bitcoin. Keďže hlavným cieľom predloženej práce je analýza transakčných dát a následná predikcia časových pásiem používateľov, je dôležité porozumieť mechanizmom, ktorými tieto dáta v sieti vznikajú, ako sa propagujú a akým spôsobom sú trvalo zaznamenávané.

V prvej časti kapitoly zadefinujeme elementárne kryptografické princípy, konkrétne hašovacie funkcie a digitálne podpisy, ktoré zabezpečujú integritu a bezpečnosť celého systému bez potreby akejkoľvek centrálnej autority. Následne sa zameriame na samotnú architektúru Bitcoinu.

Osobitná pozornosť je v texte venovaná životnému cyklu transakcie. Detailne rozoberáme mechanizmus nespotrebovaných výstupov (UTXO), ktorý je fundamentálny pre sledovanie toku transakcií, prechod transakcie dočasnou pamäťou uzlov (memory pool) a jej definitívne potvrdenie prostredníctvom procesu ťaženia (mining). Záver kapitoly je venovaný analýze vnútorných architektúr centralizovaných búrz.

1.1 Kryptografia a kryptomeny

Všetky meny potrebujú nejaký spôsob kontroly a dodržiavanie určitých bezpečnostných pravidiel, aby sa zabránilo podvodom. Pri fiat menách je táto kontrola vykonávaná centrálnymi bankami, ktoré kontrolujú ponuku meny a zabezpečujú ochranu proti falšovaniu. Pri kryptomenách musí byť tiež zabezpečená bezpečnosť aj napriek tomu, že tu nevystupuje žiadna tretia organizácia. Namiesto toho kryptomeny, ako už naznačuje slovo, využívajú kryptografiu na vytvorenie mechanizmu bezpečných pravidiel.

Kryptografia je veda o šifrovaní informácií, ktorá je notoricky známa využívaním pokročilých matematických metód a náročnosťou na pochopenie. V tejto diplomovej práci nebudeme rozoberať jednotlivé kryptografické techniky. Iba si v jednoduchosti predstavíme dve, ktoré sú kľúčové pri tvorbe kryptomien.

V tejto časti si predstavíme základné koncepty kryptografie, ktoré budeme využívať pri technickom vysvetľovaní základného fungovania kryptomeny Bitcoin. Pri vypracovávaní informácií sme vychádzali najmä z [3].

1.1.1 Hašovacia funkcia

Prvou z nich je hašovacia funkcia. Je to matematická funkcia, ktorá vytvára digitálny identifikátor pre rôzne dáta. Všeobecnú hašovaciu funkciu môžeme definovať nasledovne:

Definícia 1.1 *Všeobecná hašovacia funkcia je funkcia, ktorej úlohou je previesť ľubovoľne dlhý reťazec (ďalej označovaný ako s) na kratší reťazec, hash (neskôr označovaný ako v), ktorý má fixnú, preddefinovanú dĺžku. V matematickom vyjadrení:*
$h : \{0,1\}^* \to \{0,1\}^n$, *kde:*
$\{0,1\}^*$ *je množina všetkých reťazcov v dvojkovej sústave s ľubovoľnou dĺžkou*
$\{0,1\}^n$ *je množina všetkých reťazcov v dvojkovej sústave s dĺžkou n*
Navyše predpokladáme, že všeobecná hašovacia funkcia má nasledovné vlastnosti:

(i) *rovnaké vstupy vždy vygenerujú rovnaký výstup*

(ii) *malá zmena vstupných dát výrazne zmení výsledok*

(iii) *výstup je možné efektívne vypočítať z ľubovoľného vstupu*

Pre potreby kryptomien však takáto všeobecná funkcia nie je postačujúca. Vyžadujeme navyše, aby bola kryptograficky bezpečná. K definícii kryptografickej hašovacej funkcie pridáme ešte nasledovné vlastnosti:

- **Silná bezkolíznosť**: Hašovacia funkcia h sa nazýva silno bezkolízna, ak je výpočtovo takmer nemožné[1] nájsť dva vstupné reťazce (s_1 a s_2), pričom $s_1 \neq s_2$ také, aby $h(s_1) = h(s_2)$

- **Jednocestnosť**: Hašovacia funkcia h sa nazýva jednocestná, ak je výpočtovo takmer nemožné pri zadanom haši v nájsť vstupný reťazec s taký, že $h(s) = v$.

[1]Pod pojmom výpočtovo takmer nemožné rozumieme štatisticky nepravdepodobné, že by sme mohli výsledok dosiahnuť náhodným hádaním. Teda teoreticky je možné uspieť pomocou náhodného hádania hodnôt, ale nie je to možné v konečnom časovom horizonte.

- **Slabá bezkolíznosť**: Hašovacia funkcia h sa nazýva slabo bezkolízna, ak je výpočtovo takmer nemožné zo zadaného vstupného reťazca s_1 nájsť iný vstupný reťazec s_2 taký, že platí $h(s_1) = h(s_2)$.

1.1.2 Digitálny podpis

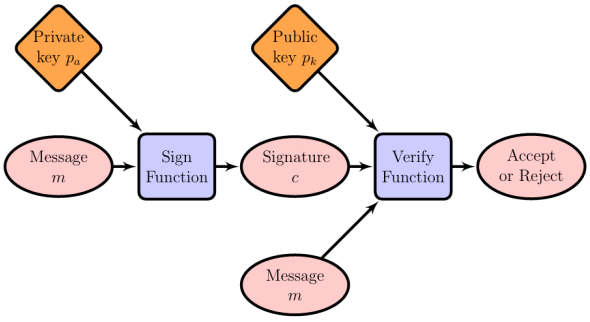
Druhá kľúčová kryptografická technika používaná pri kryptomenách je digitálny podpis. Hlavnými úlohami digitálneho podpisu sú:

(i) **Overenie pravosti**: prijímateľ vie jednoznačne overiť, že správa prišla od odosielateľa

(ii) **Nepopierateľnosť**: odosielateľ nevie poprieť odoslanie správy

(iii) **Integrita**: dôkaz o tom, že správa nebola pozmenená

Prvý krok implementácie digitálneho podpisu v praxi zahrňuje proces **generovania verejného kľúča**, kde dvojica kľúčov (verejný a súkromný) je vygenerovaná tak, aby spĺňala zadané vlastnosti. Po generácii verejného kľúča nasleduje už aplikácia digitálneho podpisu priamo na správy. Proces fungovania digitálneho podpisu je znázornený na Obr. 1. **Vytvorenie podpisu** sa vykonáva prostredníctvom „podpisovej funkcie“, ktorá kombináciou správy a súkromného kľúča odosielateľa podpíše správu. Prijímateľ následne obdrží podpísanú správu, ktorú pred prijatím verifikuje. **Verifikácia** je proces, kde sa pomocou pôvodnej správy a verejného kľúča overí, či podpis bol vytvorený príslušným súkromným kľúčom. Funkcie podpísania aj verifikácie sú verejné informácie.



Obr. 1: Proces digitálneho podpisovania správ (Zdroj: [3])

Konkrétne Bitcoin využíva digitálny podpis, ktorý sa nazýva ECDSA (Elliptic Curve Digital Signature Algorithm). Tento podpis je schválený americkou vládou a prešiel mnohými analýzami preto je považovaný za bezpečný. Podrobné detaily fungovania nebudeme v tejto práci rozoberať, lebo matematická náročnosť by si zabrala veľký rozsah.

1.2 Bitcoin a jeho architektúra

Bitcoin bol vynájdený v 2008 predstavením článku „Bitcoin: A peer-to-Peer Electronic Cash System“ [14], napísaného pod pseudonymom Satoshi Nakamoto. Nakamoto skombinoval desiatky rokov výskumu v oblasti kryptografie a distribuovaných systémov, ktoré vytvorili základ tohto elektronického platobného systému, umožňujúceho každému vytvoriť konto a posielať ľubovoľnú hodnotu komukoľvek na svete. Bol vytvorený ako alternatíva k súčasnému finančnému systému, v ktorom centrálne banky ovplyvňujú infláciu regulovaním objemu peňazí a neexistuje iná možnosť, iba dôverovať bankám.

Bitcoin umožňuje uchovávať a prenášať peňažné jednotky Bitcoin prostredníctvom transakcií v distribuovanej sieti. **Protokol Bitcoin** je dostupný ako **Bitcoin Core** softvér s otvoreným zdrojovým kódom (open-source), voľne stiahnuteľný a spustiteľný na širokej škále zariadení. Po prvom spustení programu sa najskôr pripojí k počítačom so spusteným rovnakým programom a začnú zdieľať súbor s vaším počítačom. Tento súbor sa nazýva **blockchain** a obsahuje zoznam všetkých doterajších transakcií. V momente iniciovania novej transakcie jeden počítač, na ktorom beží bitcoinový protokol (**uzol**) ju zachytí, overí jej platnosť a uloží do svojej dočasnej pamäte nazývanej memory pool, odkiaľ ju distribuuje ostatným účastníkom siete. V približne 10 minútových intervaloch náhodný uzol v sieti pridá transakcie zo svojej pamäti (memory poolu) do blockchainu a zazdieľa túto zmenu s ostatnými účastníkmi. Proces pridávania transakcií do blockchainu sa nazýva ťaženie (mining), ide o výpočtovo náročnú súťaž, ktorej výsledkom je dosiahnutie globálneho konsenzu a definitívne potvrdenie transakcií. Výsledkom je decentralizovaná peer-to-peer sieť.

Vyššie sme v krátkosti opísali základnú architektúru Bitcoinu, teraz si na príklade začínajúceho používateľa vysvetlíme bližšie detaily získavania, odosielania a prijí-

mania Bitcoinov ako aj podrobnejšie detaily fungovania celého systému. Budeme sa zameriavať najmä na to, čo je potrebné ďalej pre našu analýzu a predikciu časových pásiem. Takýmto konceptom vysvetlenia základnej teórie sme sa inšpirovali v knihe [2], ktorá spolu so zdrojmi [14] a [13] tvorí hlavnú inšpiráciu v tejto podkapitole.

1.2.1 Klasifikácia a mechanizmus bitcoinových peňaženiek

Prvým krokom pre interakciu s protokolom Bitcoin je vytvorenie digitálnej peňaženky (wallet). Existuje množstvo rôznych delení peňaženiek, v tejto práci ich rozdelíme z hľadiska bezpečnosti a správy súkromných kľúčov na softvérové (Hot Wallets) a hardvérové (Cold Wallets).

Softvérová peňaženka

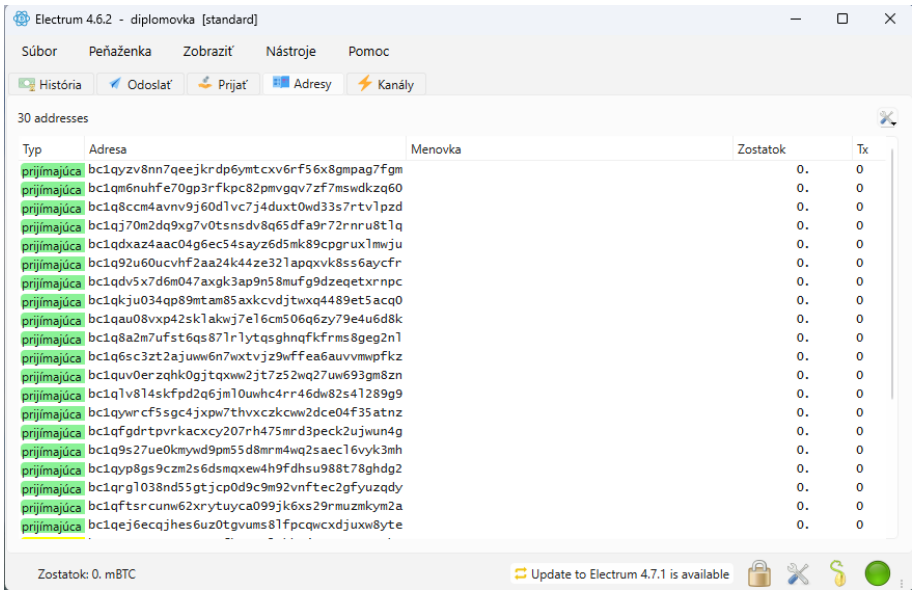
Predstavujú aplikácie spustiteľné na zariadeniach s priamym prístupom k internetu (počítače, smartfóny). Ich hlavnou výhodou je možnosť rýchlo a ľahko prijímať aj odosielať transakcie. Kritickou nevýhodou je však expozícia súkromných kľúčov online prostrediu, čo zvyšuje riziko ich získania prostredníctvom malvéru alebo phishingových útokov. V praxi sa tento typ peňaženiek považuje za vhodný, pokiaľ používateľ dodržiava základné princípy digitálnej bezpečnosti.

Hardvérová peňaženka

Ide o špecializované fyzické zariadenie, navrhnuté na izolovanú správu súkromných kľúčov. Ich architektúra je postavená na využití zabezpečeného hardvérového modulu, ktorý zaručuje, že súkromné kľúče nikdy neopúšťajú zariadenie v nezašifrovanej podobe. Hoci sa zariadenie pripája k počítaču s internetovým pripojením za účelom autorizácie transakcie, samotný proces digitálneho podpisovania prebieha vo vnútri hardvérovej peňaženky. Vďaka tomuto procesu súkromné kľúče nie sú nikdy priamo vystavené online prostrediu, čím sa eliminuje riziko odcudzenia.

V tejto práci analyzujeme príklad tvorby a fungovania softvérovej peňaženky prostredníctvom aplikácie **Electrum**. Na Obr. 2 je znázornené rozhranie peňaženky s vygenerovaným súborom adries. Tieto adresy slúžia na prijímanie Bitcoinov. Jed-

nou z hlavných funkcií peňaženiek je práve manažovanie týchto adries. V súlade s odporúčaniami Satoshi Nakamota v [14] je pre každú prichádzajúcu transakciu generovaná unikátna adresa, čo prispieva k vyššej miere anonymity používateľa. Hoci



Obr. 2: Rozhranie peňaženky Electrum so zoznamom vygenerovaných adries

sa v používateľskom rozhraní narába primárne s adresami, pri samotnej konštrukcii transakcií sa využíva digitálny podpis a s ním súvisiace kľúče, ktoré sme predstavili v časti 1.1.2. Adresy v tomto procese plnia skôr reprezentačnú funkciu a slúžia ako cieľový bod platby.

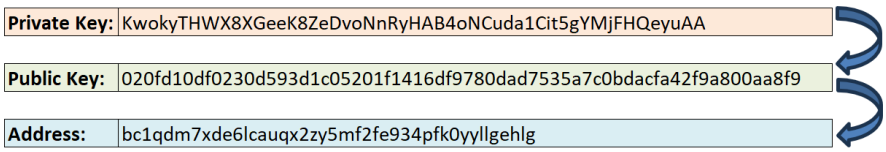
Každá adresa vzniká odvodením od súkromného kľúča. Vo všeobecnosti môžeme teda tvrdiť, že peňaženka má na starosti skôr generovanie a bezpečnú správu súkromných kľúčov ako adries. Súkromný kľúč predstavuje náhodne vygenerované 256 bitové číslo v dvojkovej sústave z rozsahu 1 až 2^{256}. Pre lepšiu predstavu ho možno definovať ako unikátne heslo, ktoré majiteľovi umožňuje autorizovať odosielanie platieb.

K tomuto kľúču sa prostredníctvom jednosmernej matematickej funkcie pomocou násobenia na eliptickej krivke (secp256k1) odvodzuje verejný kľúč. Vzhľadom na zameranie tejto práce nebudeme analyzovať tento proces detailnejšie. Dôležité je však uvedomiť si jednosmernosť tejto matematickej operácie. Zo súkromného kľúča je kedykoľvek možné vypočítať jednoznačný verejný, avšak opačný proces nie je realizovateľný. Verejný kľúč si možno predstaviť ako číslo účtu, ktoré poskytujeme

odosielateľovi.

Samotná adresa je následne odvodená prostredníctvom hašovacej funkcie. Jej hlavným významom je transformácia dát do kompaktnejšej a prehľadnejšej podoby určenej pre koncového používateľa. Pri manuálnej konštrukcii transakcie nie je potrebné adresu používať, avšak adresa bola navrhnutá pre zjednodušenie procesu posielania platieb cez klientske aplikácie. Na Obr. 3 je znázornený tento proces pre prvú adresu vygenerovanú peňaženkou Electrum. Zobrazené hodnoty predstavujú číselné údaje v rôznych sústavách (šestnástková sústava, Base58).

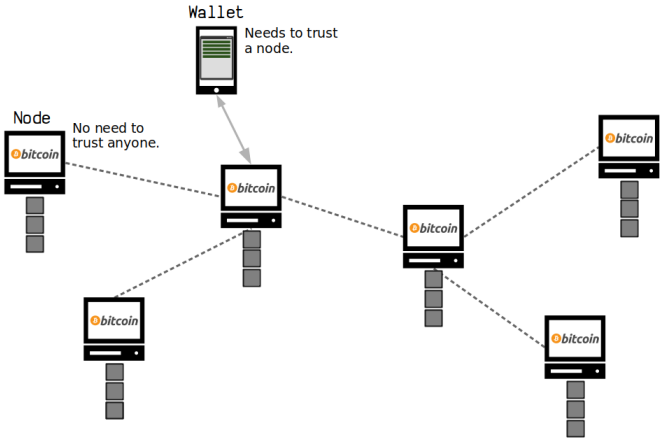
Okrem správy kľúčov je nezanedbateľnou funkciou peňaženky vytváranie transakcií a ich následná propagácia do siete Bitcoin.



Obr. 3: Proces transformácie kryptografických kľúčov

V prípade prevádzky plnohodnotného vlastného uzla na lokálnom hardvéri je možné toto zariadenie využiť aj ako peňaženku na generovanie kľúčov a konštrukciu transakcií. Tento prístup je určený pre pokročilejších používateľov, pričom jeho hlavná výhoda spočíva v eliminácii potreby dôverovať inému uzlu pri odosielaní transakcií (Obr. 4). Jeden z príkladov problému dôvery pri bežných peňaženkách spočíva v tom, že pri zisťovaní zostatkov sa peňaženka musí dopytovať externého uzla na všetky adresy patriace danému používateľovi. Prevádzkovateľ cudzieho uzla by mohol tento súbor adries priradiť k jednému používateľovi a tým ho deanonymizovať. V takomto prípade by bola podstata generovania nových adries pre každú transakciu z hľadiska súkromia neúčinná.

Naopak, pri využití vlastného uzla sa validácia transakcií ako aj overovanie zostatku spolieha na osobný uzol a jeho kópiu blockchainu. Nevýhodou tohto riešenia je vysoká náročnosť na hardvérové prostriedky, nakoľko vlastný uzol vyžaduje stiahnutie a neustálu verifikáciu celej histórie transakcií (dnes približne 830 GB). Alternatívnym kompromisom pre pokročilých používateľov je manuálna konfigurácia peňaženky na pripojenie k vopred overeným a dôveryhodným uzlom.



Obr. 4: Schéma interakcie medzi peňaženkou a uzlom z hľadiska informačnej bezpečnosti (Zdroj: [13])

1.2.2 Metódy nadobudnutia bitcoinu

Získanie prvých bitcoinov predstavuje pre nových používateľov odlišný proces v porovnaní s konvenčnými menovými operáciami v bankovom sektore. Keďže bitcoin funguje nezávisle od centrálnych autorít, jeho nadobudnutie nie je možné v žiadnej centrálnej inštitúcii. Proces jeho nadobudnutia je však možný nasledujúcimi spôsobmi:

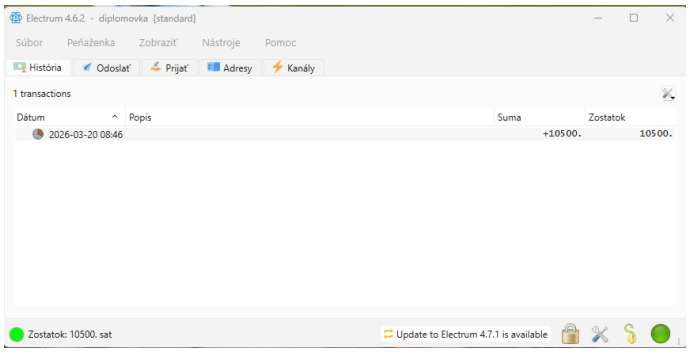
1. **Priama výmena:** Ide o nákup priamo od iného držiteľa bez prostredníka. Tento spôsob najlepšie odzrkadľuje pôvodnú ideu bitcoinu ako decentralizovaného platobného systému. K takejto výmene dochádza buď v rámci osobnej siete kontaktov alebo na špecializovaných komunitných podujatiach (príklady takýchto podujatí na Slovensku možno nájsť na [20]).

2. **Bitcoinové bankomaty:** Fyzické zariadenia umožňujúce nákup bitcoinu za hotovosť alebo prostredníctvom platobnej karty. Výhodou je relatívna rýchlosť a dostupnosť, nevýhodou sú spravidla vyššie transakčné poplatky v porovnaní s online burzami.

3. **Prijatie platby za tovary a služby:** Používateľ môže nadobudnúť bitcoin ako protihodnotu za ekonomickú činnosť, čím sa digitálne aktívum stáva priamym výmenným prostriedkom.

4. **Burzy:** Platformy, ktoré umožňujú nákup bitcoinu prostredníctvom bankového prevodu alebo karty. Dôležitým aspektom je, že pri držaní prostriedkov na burzovom účte používateľ reálne nedisponuje súkromnými kľúčmi. V tomto stave má plnú kontrolu nad aktívami burza, čo nesie riziko straty prostredníctvom hackerského útoku alebo platobnej neschopnosti platformy. Preto sa v rámci bezpečnosti odporúča zakúpený bitcoin okamžite previesť na peňaženku. Viac o burzách uvádzame v časti 1.2.6.

Najbežnejším spôsobom nadobudnutia je využitie búrz. V rámci praktickej ukážky sme zrealizovali nákup a následný prevod objemu 10 500 satoshi (pričom 1 BTC = 10^8 sats) z burzy Binance na vytvorenú Electrum peňaženku (Obr. 5).



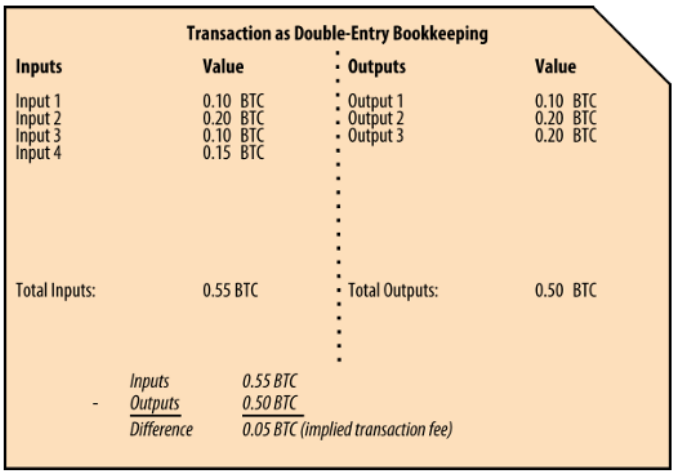
Obr. 5: Prevod z burzy Binance na Electrum peňaženku

1.2.3 Mechanizmus realizácie a propagácie transakcií

Po nadobudnutí bitcoinu do osobnej správy v peňaženke je možné realizovať transakcie v rámci bitcoinovej siete. V tejto časti zahrnieme príklad reálnej transakcie za platbu v kaviarni. Na vizualizáciu procesov a sledovanie transakcií v sieti Bitcoin budeme využívať tzv. blockchainové prehliadače (blockchain explorers). Sú to webové aplikácie, ktoré slúžia ako vyhľadávacie nástroje v transakčnej databáze (blockchain, memory pool) a umožňujú sledovať stavy adries. Pre účely tejto práce sme zvolili rozhranie dostupné na https://www.blockchain.com/explorer.

Z koncepčného hľadiska možno bitcoinovú transakciu prirovnať k záznamu v účtovnej knihe. Každá transakcia pozostáva z jedného alebo viacerých vstupov (strana Má dať) a výstupov (strana Dal). Na rozdiel od klasického účtovníctva sa súčet vstupov a výstupov nerovná. Rozdiel medzi nimi predstavuje transakčný poplatok.

Tento poplatok získa počítač/uzol za schválenie a zahrnutie transakcie do bloku a jeho pridanie do blockchainu. Príklad takejto štruktúry ilustruje Obr. 6.



Obr. 6: Transakcia zobrazená ako účtovná kniha v podvojnom účtovníctve (Zdroj: [2])

UTXO (Unspent transaction Output)

Bitcoin nevyužíva model zostatkov na účtoch ako bežné banky, ale model nespotrebovaných výstupov (UTXO). Vstupy do transakcie neobsahujú priamo adresu, ale referenciu na výstup z predchádzajúcej transakcie, ktorý ešte nebol spotrebovaný. Tieto výstupy si možno predstaviť ako uzamknuté depozitné schránky v banke, v ktorých sa nachádza určitá hodnota.

Pri vytváraní transakcie peňaženka vyberie vhodné depozitné schránky, ktorých celková hodnota je vyššia alebo rovná požadovanej sume a pomocou digitálneho podpisu ich „odomkne“. Následne sa vytvoria nové výstupy (depozitné schránky), ktoré sa uzamknú podmienkami prislúchajúcimi príjemcom. Dôležitou vlastnosťou je, že každé UTXO je možné využiť iba raz. Ak je hodnota vstupu vyššia než platba, zvyšná časť sa vracia odosielateľovi vo forme nového nespotrebovaného výstupu. Celý blockchain je teda možné chápať ako úložisko pre všetky depozitné schránky (výstupy).

Praktický príklad: Platba v kaviarni

Modelovú situáciu platby v hodnote 5000 sats ilustruje Obr. 7, kde je znázornený blok z kaviarne vo forme QR kódu. Po naskenovaní kódu peňaženka automaticky pripraví transakciu. Najskôr si však v pozadí overí, či má používateľ dostatočnú sumu



Obr. 7: Blok z kaviarne

nespotrebovaných výstupov. Keďže boli na adresu *bc1qdm7...* prevedené bitcoiny zo zmenárne Binance, očakávame existenciu jedného UTXO. Prostredníctvom prehliadača sme získali výstup k UTXO na tejto adrese (Obr. 8). Peňaženka vykonáva niečo podobné pre všetky adresy patriace používateľovi. Na zmienenom obrázku vidíme nasledovné údaje:

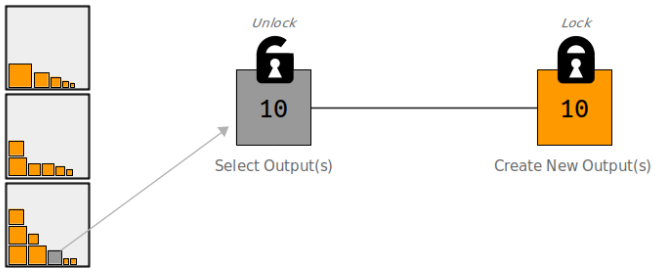
- **tx_hash_big_endian:** Väčšinou pri BTC označované ako TXID a ide o unikátny identifikátor transakcie, v ktorej nespotrebovaný výstup vznikol. TXID je vo všeobecnosti generovaný hašovacou funkciou aplikovanou na niektoré špecifické časti dát o transakcii.
- **tx_output_n:** Index (VOUT), ktorý určuje poradie výstupu v danej transakcii.
- **script:** Uzamykací mechanizmus pre výstupy transakcií. Jeho tvar sa môže líšiť v závislosti od používaného typu adresy. V tomto prípade obsahuje po prvých 4 číslach hašovaný verejný kľúč prislúchajúci k adrese.
- **value:** Nominálna hodnota nespotrebovaného výstupu v satoshi.


```
{'notice': '',
 'unspent_outputs': [{'tx_hash_big_endian': '7a9db8fcbfabc8799d179e85507f5c035885fad4455048ef03069396e99a527a',
   'tx_hash': '7a529ae996930603ef485045d4fa8558035c7f50859e179d79c8abbffcb89d7a',
   'tx_output_n': 6,
   'script': '00146efc66e75fc778032844a6d2a4e4b1a85367909f',
   'value': 10500,
   'value_hex': '2904',
   'confirmations': 16,
   'tx_index': 4303846557012576}]}
```

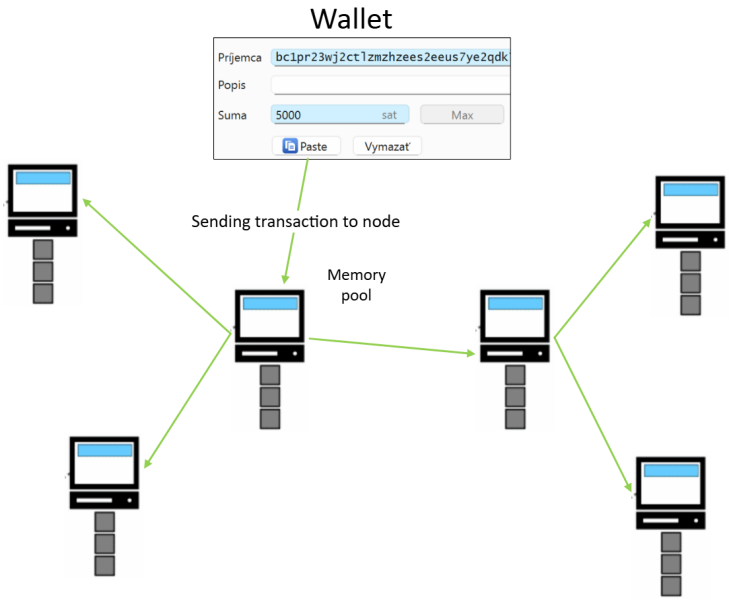

Obr. 8: Nespotrebované výstupy k adrese: *bc1qdm7xde6lcauqx2zy5mf2fe934pfk0yyllgehlg*

Po odoslaní transakcie z peňaženky Electrum sa automaticky zvolí nespotrebovaný výstup alebo viacero tak, aby ich súčet pokryl sumu a poplatok. Peňaženka

tieto vstupy odomkne a vytvorí nové nespotrebované výstupy (UTXOs), ktoré zamkne poskytnutým verejným kľúčom prijímateľa (proces znázornený na Obr. 9). Táto transakcia sa propaguje do siete a ukladá do memory poolu uzlov. Proces propagácie novej transakcie je znázornený na Obr. 10. Po približne 10 minútach jeden z uzlov v sieti vytvorí zo svojho memory poolu blok a pridá ho do blockchainu. Nový blok sa zazdieľa ostatným uzlom, ktoré ho overia a tiež zaradia do svojho blockchainu. Po tom, ako je transakcia zahrnutá do nového bloku, je potvrdená a platba je vykonaná. Po potvrdení je transakciu už takmer nemožné zmeniť.



Obr. 9: Proces tvorby vstupov a výstupov transakcie (Zdroj: [13])



Obr. 10: Propagácia novej transakcie (Zdroj: [13] + vlastné úpravy)

Analýza potvrdenej transakcie

V tejto časti si analyzujeme štruktúru potvrdenej transakcie prostredníctvom prehliadača blockchain.info (Obr. 11). Hoci sa formáty výstupov v rôznych prehlia-

dačoch a klientskom softvéri (Bitcoin Core) môžu líšiť, tento konkrétny zdroj sme zvolili najmä preto, že neskôr budeme tento prehliadač využívať aj ako zdroj dát našej analýzy.


```
{
  "txid": "d874bae7eb37f5fa05934167de2ff90c3a1f1977c9c0089dfe49cf8e85787940",
  "size": 234,
  "version": 2,
  "locktime": 941529,
  "fee": 200,
  "inputs": [
    {
      "coinbase": false,
      "txid": "7a9db8fcbfabc8799d179e85507f5c035885fad4455048ef03069396e99a527a",
      "output": 6,
      "sigscript": "",
      "sequence": 4294967293,
      "pkscript": "00146efc66e75fc778032844a6d2a4e4b1a85367909f",
      "value": 10500,
      "address": "bc1qdm7xde6lcauqx2zy5mf2fe934pfk0yyllgehlg",
      "witness": [
        "30440220475d24bea7cad021c6c115b3787f7d1a54e2afc790203d582bb37f0a37c3e08402201c2a42b1d609b9eab902e53aa5732568e1336e7db3aaeaa7650355f1dd0fa74101",
        "020fd10df0230d593d1c05201f1416df9780dad7535a7c0bdacfa42f9a800aa8f9"
      ]
    }
  ],
  "outputs": [
    {
      "address": "bc1pr23wj2ctlzmzhzees2eeus7ye2qdk7r9yn7qdxf7njys78sg0e8ssx0mzf",
      "pkscript": "51201aa2e92b0bf8b62b8b3982b39e43c4ca80db786524fc06993e9c890f1e087e4f",
      "value": 5000,
      "spent": false,
      "spender": null
    },
    {
      "address": "bc1qx5mvctfrjfcptf0gpg0nt2u5gdrtmzhuqv0qcp",
      "pkscript": "00143536cc2d23927015a5e80a1f35ab944346bd8afc",
      "value": 5300,
      "spent": false,
      "spender": null
    }
  ],
  "block": {
    "height": 941530,
    "position": 1702
  },
  "deleted": false,
  "time": 1774079576,
  "rbf": false,
  "weight": 609
}
```

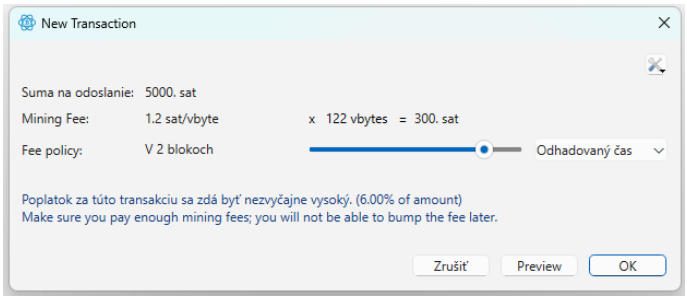

Obr. 11: Transakcia

Atribúty transakcie zobrazené v prehliadači zahŕňajú:

- **txid:** Ako sme už spomenuli ide o jedinečný identifikátor transakcie vytvorený z transakčných dát.
- **size/vSize:** Fyzická veľkosť transakcie v bajtoch (B) a virtuálnych bajtoch (vB). Hodnota vSize je kľúčová pre výpočet transakčných poplatkov a vypočítava sa ako $\frac{weight}{4}$.
- **fee:** Transakčný poplatok, ktorý získa uzol, ktorý blok s danou transakciou vyťaží. Väčšina peňaženiek umožňuje nastaviť výšku poplatku v jednotkách $sats/vB$ (Obr. 12). Najdôležitejšou funkciou tohto poplatku je jeho súvis s rýchlosťou schválenia transakcie do bloku (väčšinou transakcia s vyšším poplatkom bude skôr schválená). Väčšina peňaženiek teda nenastavuje priamo

výšku poplatku ale skôr očakávanú rýchlosť schválenia, tak ako to je aj na Obr. 12.

- **inputs:** Časť obsahujúca referencie na čerpané UTXO.
  - **txid:** Id transakcie, z ktorej sa spotrebúva výstup.
  - **output:** Poradové číslo výstupu v transakcii, z ktorej sa čerpá výstup.
  - **pkscript:** Zamykací mechanizmus výstupu.
  - **value:** Hodnota nespotrebovaného výstupu v satoshi
  - **address:** Adresa, ktorá obsahuje daný nespotrebovaný výstup.
  - **witness:** Odomykací mechanizmus, ktorý pri tomto type transakcie obsahuje digitálny podpis, ktorý vznikol tak ako sme opísali v časti 1.1.2 a zobrazili na Obr. 1 a verejný kľúč. Prostredníctvom verifikačnej funkcie uzol potvrdzuje, že verejný kľúč korešponduje s podmienkou v uzamykacom mechanizme (pkscript) a že priložený digitálny podpis matematicky potvrdzuje, že transakciu vykonal majiteľ príslušného súkromného kľúča.
- **outputs:** Táto časť zodpovedá výstupom, časti výstupu obsahujú adresu ku ktorej sa vytvárajú nové nespotrebované výstupy a uzamykací mechanizmus.
- **block:** Obsahuje index bloku (block height), v ktorom bola transakcia potvrdená ako aj poradové číslo transakcie v konkrétnom bloku. Počet následne vyťažených blokov určuje počet konfirmácií, čo zvyšuje mieru nezvratnosti transakcie. Informácia o počte konfirmácií často vystupuje v peňaženkách.
- **time:** Čas v sekundách, ktoré uplynuli od 1. 1. 1970. V tomto prehliadači tento čas vyjadruje zapísanie transakcie do memory poolu od bloku 725172. Pri starších blokoch tento čas reprezentuje čas schválenia bloku, v ktorom sa nachádza transakcia. Pre účely tejto diplomovej práce je tento údaj kritický, pretože v prípade moderných blokov tento prehliadač uvádza čas prijatia do mempoolu, čo predstavuje presnejšiu aproximáciu reálnej aktivity používateľa než čas vyťaženia bloku.



Obr. 12: Nastavenie poplatku za transakciu v peňaženke Electrum

Časti transakcie, ktoré sme opísali, nemusia byť vždy úplne totožné. Je to spôsobené tým, že existuje viacero typov transakcií podľa toho, z akého typu adries sa čerpajú výstupy. Pre potreby tejto diplomovej práce nebudeme tieto typy ďalej špecifikovať. Jedným z dôvodov ich vzniku bolo napríklad zníženie poplatku za transakcie.

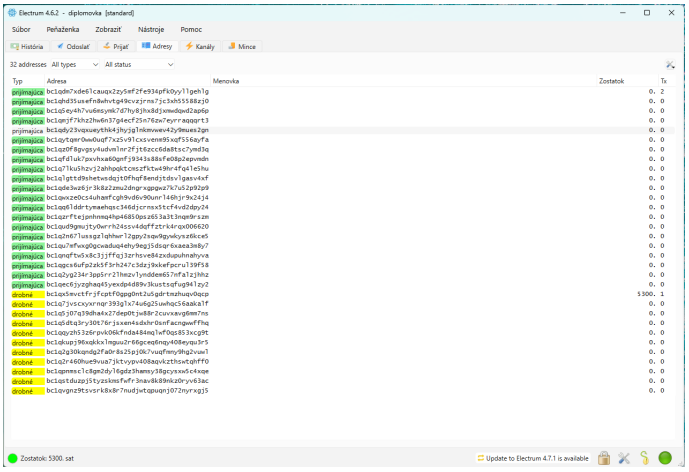
Adresa pre zvyšok

Môžeme si všimnúť, že transakcia, ktorú sme uviedli na Obr. 11 má jeden vstup a 2 výstupy. Pri analýze adries, na ktorých boli vytvorené UTXOs je zrejmé, že ani jedna adresa nie je zhodná so vstupnou (Obr. 13).

V modelovom príklade platby v kaviarni (5000 sats) transakcia spotrebováva UTXO v hodnote 10500 sats, a preto peňaženka vytvorila tzv. adresu pre zvyšok (change adrress), kam sa uloží nespotrebovaná časť vstupu. Z dôvodu zachovania anonymity väčšina moderných peňaženiek generuje pre každý zvyšok novú adresu. Externý pozorovateľ bez dodatočných informácií nedokáže exaktne určiť, ktorá z výstupných adries patrí príjemcovi a ktorá slúži na vrátenie zvyšku odosielateľovi. V našom prípade však vieme identifikovať druhú adresu v poradí ako adresu pre zvyšok, čo potvrdzuje aj zostatok v peňaženke Electrum (Obr. 14).



Obr. 13: Prehľad vstupných a výstupných adries pri platbe v kaviarni



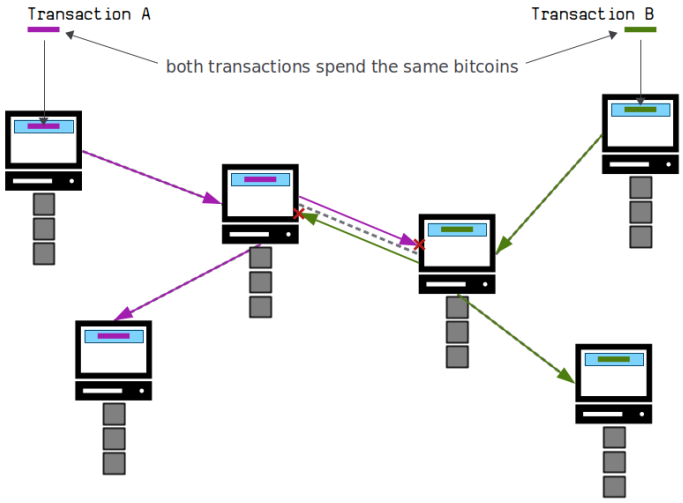
Obr. 14: Prehľad vytvorených adries pre prijímanie BTC aj adries pre zvyšok peňaženkou Electrum. Zostatok pri adrese pre zvyšok potvrdzuje, že peňaženka vytvorila nový UTXO v transakcii pri platbe v kaviarni.

1.2.4 Memory pool

Ako sme uviedli v úvode tejto podkapitoly, memory pool slúži ako čakáreň pre nepotvrdené transakcie. Po odoslaní sa každá transakcia najskôr propaguje do mempoolov jednotlivých uzlov, kde čaká na validáciu a následné zahrnutie do bloku. Protokol Bitcoin je nastavený tak, že nový blok je vyťažený v priemere každých 10 minút. Naskytá sa teda otázka, prečo transakcie nie sú potvrdzované okamžite, čím by sa eliminovala čakacia doba. Hlavným bezpečnostným dôvodom existencie memory poolu je ochrana pred konfliktnými transakciami (double spending).

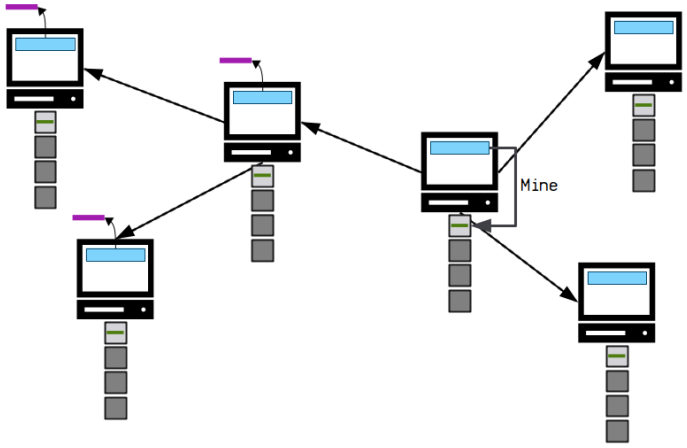
Predstavme si situáciu, kedy sa útočník pokúsi odoslať 2 transakcie, ktoré spotrebúvajú rovnaké UTXOs. Tieto transakcie by boli vložené do rozdielnych častí siete (iné uzly) v rovnakom čase. Kvôli sieťovej latencii by časť uzlov prijala do svojho memory poolu prvú transakciu a časť uzlov druhú. Tento stav dočasnej nekonzistencie siete ilustruje Obr. 15.

Bez existencie memory poolu a procesu ťaženia by nebolo možné určiť, ktorá transakcia má byť zapísaná do blockchainu. Memory pool však umožňuje, aby obe transakcie boli propagované, pričom konečný konflikt sa vyrieši až v momente vyťaženia nového bloku. Uzol, ktorý schváli transakcie zahrnie do bloku tú transakciu, ktorú mal v čase ťaženia vo svojom mempoole. V momente, kedy je tento nový blok propagovaný zvyšku siete, ostatné uzly aktualizujú svoj stav: prijmú nový blok,



Obr. 15: Zobrazenie pridania 2 transakcií čerpajúcich z rovnakých UTXOs a ich propagácia v sieti. (Zdroj: [13])

identifikujú v ňom potvrdenú transakciu a následne zo svojich mempoolov odstránia všetky s ňou konfliktné (duplicitné) transakcie. Tento proces zabezpečuje integritu blockchainu a jednoznačnosť transakčnej histórie (Obr. 16)



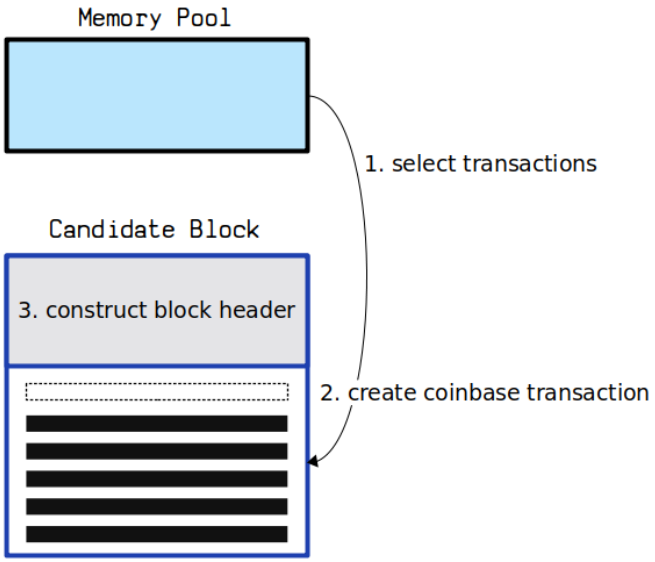
Obr. 16: Zobrazenie riešenia problému konfliktných transakcií. (Zdroj: [13])

1.2.5 Schvaľovanie transakcií - Ťaženie

V tejto časti si bližšie priblížime proces schvaľovania transakcií, označovaný ako ťaženie. Ťaženie sa označuje ako proces, pri ktorom sa uzly pokúšajú o pridanie nového bloku do blokchainu. Tento mechanizmus možno chápať ako súťaž medzi uzlami, do ktorej sa môže zapojiť ľubovoľný uzol v sieti a súťažiť o odmenu v

podobe bitcoinov.

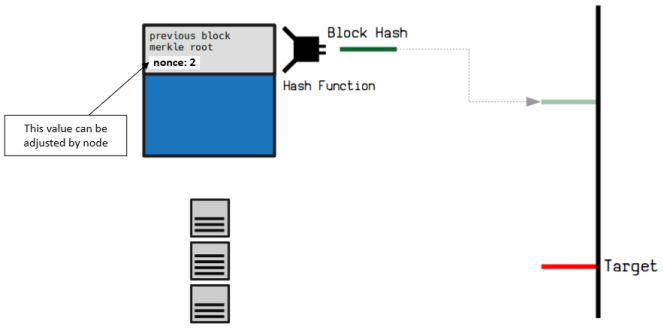
Proces ťaženia začína tvorbou kandidátskeho bloku (candidate block). Prvým krokom pri tvorbe kandidátskeho bloku je voľba transakcií z memory poolu vlastného uzla. Ťažiar prioritne volí transakcie s najvyššími transakčnými poplatkami, aby maximalizoval svoju výšku odmeny. Následne je vytvorená coinbase transakcia. Ide o špeciálny typ transakcie bez predchádzajúcich vstupov, ktorá pripisuje odmenu za vyťaženie bloku na adresu ťažiara. Poslednou fázou prípravy je vytvorenie hlavičky bloku, ktorá obsahuje metadáta sumarizujúce obsah bloku. Obsahuje 6 rôznych polí, z ktorých jedno z nich sa nazýva nonce. Toto pole predstavuje číselnú hodnotu, ktorú môže ťažiar ľubovoľne meniť. Proces tvorby kandidátneho bloku je ilustrovaný na Obr. 17.



Obr. 17: Proces tvorby kandidátskeho bloku (Zdroj: [13])

Po zostavení hlavičky dochádza k aplikácii kryptografickej hašovacej funkcie na dáta hlavičky, čím vzniká block hash. Block hash je jedinečný identifikátor bloku, ktorý predstavuje 256-bitové číslo. Súťaž spočíva v hľadaní takej hodnoty nonce, aby výsledný hash bloku bol nižší ako aktuálna cieľová hodnota (target). Tento parameter sa nastavuje raz za 2016 blokov tak, aby priemerný interval medzi blokmi zostal na úrovni 10 minút. Tento proces je zobrazený na Obr. 18.

Ak ťažiar nájde platné riešenie, blok je okamžite propagovaný do siete. Ako kompenzácia za vynaložené úsilie sa poskytuje bloková odmena (block reward). Táto



Obr. 18: Tvroba block hashu a porovnanie hashu so stanovenou cieľovou hodnotou (target) (Zdroj: [13] + vlastné úpravy)

odmena pozostáva z dvoch zložiek: sumy transakčných poplatkov všetkých transakcií zahrnutých v kandidátnom bloku a generáciou nových bitcoinov. Emisia nových bitcoinov je limitovaná celkovým počtom 21 miliónov BTC a podlieha procesu tzv. halvingu. Tento proces každých 210 000 blokov znižuje odmenu na polovicu. Pôvodná odmena 50 BTC sa postupne znížila až na súčasných 3.125 BTC.

1.2.6 Burzy a ich architektúra

Jedným zo spôsobov nadobudnutia bitcoinov výmenou za bežné fiat meny je využitie búrz. Väčšina z nich funguje na princípe, že používateľ nemá prístup k súkromným kľúčom. Bitcoiny po výmene figurujú iba ako číselný zostatok na používateľskom konte, pričom používateľ nemá priamo vytvorenú vlastnú adresu. V tejto časti si priblížime, ako tieto burzy manažujú adresy a ako fungujú transakcie, ktoré vykonávajú v rámci siete Bitcoin. Existujú však aj decentralizované burzy, ktoré zabezpečujú výmenu aktív priamo na adresy, ktorých kľúče sú pod výhradnou správou používateľa.

Najväčšou nevýhodou centralizovaných búrz je riziko ich napadnutia hackermi a následné ohrozenie uložených prostriedkov. Ďalšou nevýhodou je často vyžadovaná registrácia a identifikácia používateľa prostredníctvom dokladov (proces KYC). Identifikácia v prípade búrz pôsobí ako „trade-off“ medzi anonymitou a bezpečnosťou. Burzy vyžadujúce striktnú identifikáciu sú spravidla považované za bezpečnejšie, no stráca sa pri nich anonymita.

Vnútorná peňaženková architektúra búrz je zvyčajne navrhnutá nasledovne:

- **Hot wallets:** Slúžia na spracovanie odchádzajúcich transakcií (výberov používateľov). Keďže sú neustále pripojené k internetu, z bezpečnostných dôvodov sa na nich nedržia príliš vysoké sumy.

- **Cold wallets:** Prevažná väčšina klientských prostriedkov je držaná práve na týchto adresách, ktoré sú izolované od internetu.

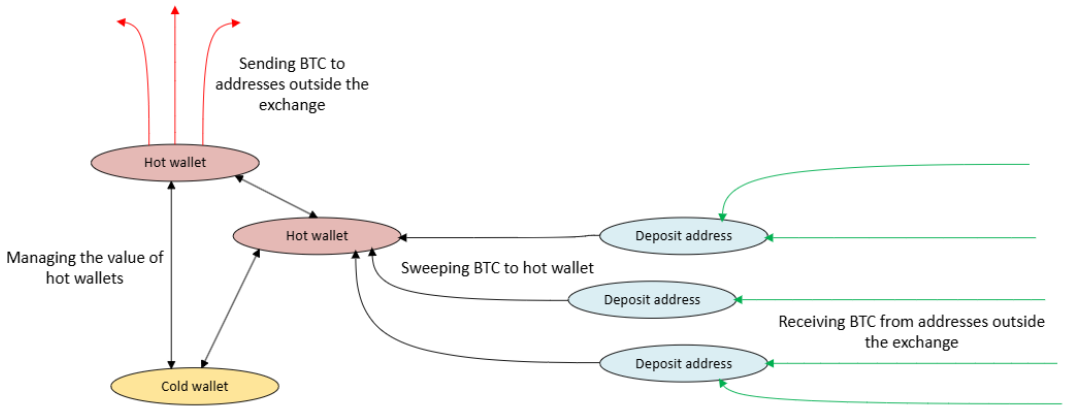
- **Deposit wallets:** Pre každého používateľa, ktorý chce vykonať prevod bitcoinov na svoj účet, je vygenerovaná unikátna adresa, ktorá zmenárni umožňuje správnu identifikáciu prichádzajúcej platby a jej priradenie ku konkrétnemu kontu.

Pôvodne sme dáta pre analýzu časových pásiem chceli zbierať práve z búrz, ktoré uplatňujú určité geografické reštrikcie (ich používanie je obmedzené na obyvateľov konkrétnych krajín). Hoci sme od tohto prístupu nakoniec upustili, pri skúmaní konkrétnych búrz sme si všimli rôzne postupy pri manažovaní prostriedkov.

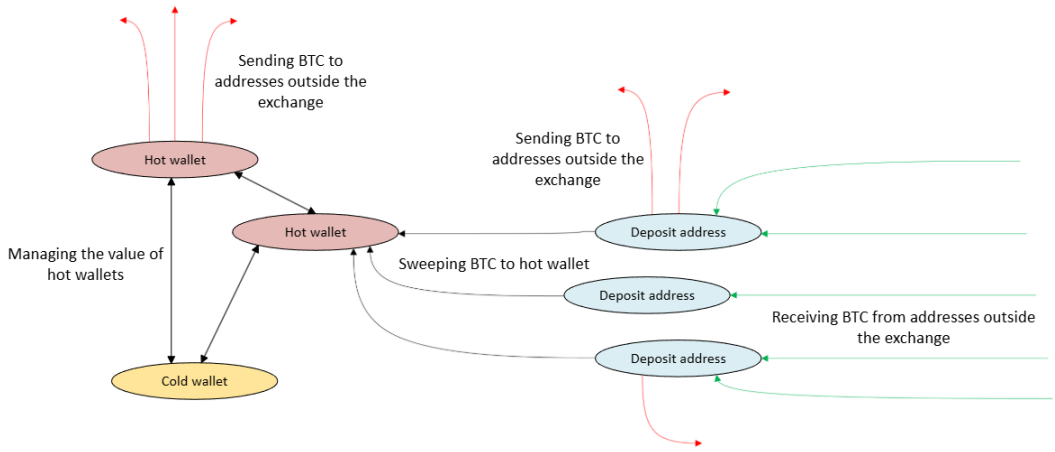
Prvý systém prevodov v rámci burzy je znázornený na Obr. 19. Depozitné adresy, sú vytvorené výhradne na prijímanie a identifikáciu vlastníctva prostriedkov. Následne sú takmer okamžite, alebo s miernym oneskorením, všetky prostriedky presunuté z depozitných adries na hot wallets. Tie sú primárne používané na odosielanie transakcií na externé adresy mimo burzy. Pretože sú pripojené k internetu a nesú určité riziko napadnutia, burzy neustále ich objem balansujú medzi cold wallets a hot wallets tak, aby na hot wallets nebolo príliš málo ani príliš veľa likvidity. Cieľová adresa pre zber prostriedkov z depozitných adries je často iná ako na samotné odosielanie transakcií, hoci to nie je pravidlom. Takýto prístup využívajú napríklad burzy Bitbank, Bithumb, CoinDCX a Coincheck.

Ďalší systém vnútornej architektúry burzy je ilustrovaný na Obr. 20. Oproti prvému sa líši tým, že depozitné adresy sa niekedy využívajú na posielanie prostriedkov priamo von z burzy. Vynechávajú sa tak medzikroky presunov na hlavné hot wallets, čím sa šetria transakčné poplatky, avšak deje sa to na úkor prehľadnosti a jednoznačnosti toku. Tento systém sme identifikovali pri burze Bitkub.

Tretí odpozorovaný systém je znázornený na Obr. 21. Pri tomto systéme nefigurujú vôbec hot wallets a cold wallets. Prichádzajúce transakcie smerujú na depozitné

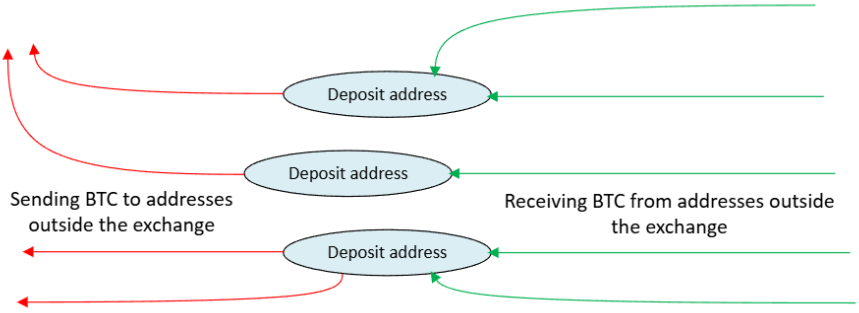


Obr. 19: Hierarchický systém s presunom z depozitných adries na Hot Wallets



Obr. 20: Hybridný systém s priamym odosielaním z depozitných adries

adresy, no pri odosielaní sa už nedodržiavajú fixné pravidlá. Často sa kombinujú nespotrebované výstupy z veľkého množstva rôznych adries, pravdepodobne s cieľom maximalizovať úsporu na transakčných poplatkoch. Takýto spôsob manažovania nám prišiel ako najmenej prehľadný a značne znemožňuje dokazovanie dostatočnosti rezerv na krytie záväzkov v prípade výberov bitcoinov. Takýto systém sme objavili pri burze Mercado Bitcoin.



Obr. 21: Systém manažovania transakcií bez jasnej hierarchie peňaženiek

2 Dáta

Primárnym cieľom tejto diplomovej práce je návrh a implementácia klasifikačného modelu schopného predikovať časové pásmo, resp. geografický región, z ktorého pochádzajú jednotliví používatelia realizujúci transakcie v sieti Bitcoin. Základným predpokladom pre úspešný vývoj takéhoto modelu je dostupnosť a konzistencia dát s vysokou výpovednou hodnotou. V tejto kapitole detailne popíšeme metodiku zberu surových dát, proces ich čistenia a následnú tvorbu finálneho datasetu, ktorý slúži ako vstup do analytickej časti práce.

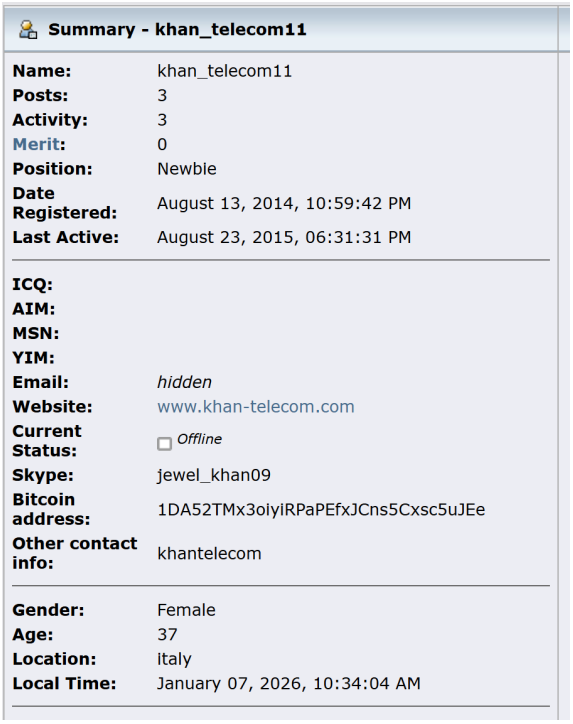
2.1 Krok 1 - bitcoinové adresy s polohou

Ako vyplýva z predchádzajúcej kapitoly, protokol Bitcoin vo svojej podstate neobsahuje informácie o geografickej polohe používateľov či uzlov. Získanie relevantných údajov o lokalite entít v sieti je náročnou úlohou, hraničiacou s limitmi anonymity bitcoinového systému. Existujú však alternatívne metódy získavania týchto dát prostredníctvom externých zdrojov. V tejto práci sme aplikovali postup inšpirovaný metodikou navrhnutou v [11].

Uvedený článok sa primárne nezaoberá geografickou lokalizáciou adries, avšak v rámci experimentálnej časti autori získali bitcoinové adresy používateľov z diskusného fóra Bitcointalk [7]. Toto fórum bolo založené v roku 2009 priamo tvorcom bitcoinu, známym pod pseudonymom Satoshi Nakamoto, a predstavuje historicky prvé a najväčšie diskusné fórum zamerané na kryptomeny. V súčasnosti obsahuje databázu približne 3.7 milióna registrovaných používateľov.

Vizuálna ukážka používateľského profilu je zobrazená na Obr. 22. Ako si môžeme všimnúť profilové rozhranie umožňuje dobrovoľne vyplniť polia pre bitcoinovú adresu (**Bitcoin address**) a polohu (**Location**). Vzhľadom na dobrovoľný charakter poskytnutých údajov predpokladáme, že ak sú tieto údaje uvedené, s vysokou pravdepodobnosťou zodpovedajú skutočnosti.

Následne prebehlo systematické spracovanie používateľských profilov. Keďže platforma neumožňuje priame filtrovanie používateľov podľa vyplnených polí, bolo nevyhnutné iteratívne spracovať všetkých približne 3,7 milióna registrovaných použí-



Obr. 22: Príklad profilu používateľa na fóre Bitcointalk s vyplnenou adresou a polohou

vateľov. Každý profil je definovaný jedinečným identifikátorom, ktorý vystupuje v URL adrese, napríklad `https://bitcointalk.org/index.php?action=profile;u=366384`. Hodnota identifikátora (v tomto prípade 366384) zodpovedá poradovému číslu registrácie.

Iterovaním tohto identifikátora bolo možné proces plne automatizovať v programovacom jazyku Python. Získané dáta boli priebežne ukladané do databázovej tabuľky v systéme MySQL, ktorej štruktúra je zobrazená na Obr. 23. Pre každého používateľa sme zaznamenali nasledujúce atribúty: identifikátor používateľa (**id**), bitcoinová adresa (**address**) a lokalita (**location**).

Z celkového počtu 3.7 milióna spracovaných profilov uviedlo bitcoinovú adresu 129 058 používateľov. Z tejto podmnožiny disponovalo súčasne vyplnenou informáciou o geografickej polohe 77 561 používateľov.

id	address	location
366384	1DA52TMx3oiyiRPaPEfxJCns5Cxsc5uJEe	italy
366385	NULL	NULL
366505	1N7YN6tAQrH8dDUspyULp3kmqMXCHW58M3	Dominican Republic
366541	18PcmZ4fBYMz8f26vQG4V13k7bMUTakR9Z	Ukraine
NULL	NULL	NULL

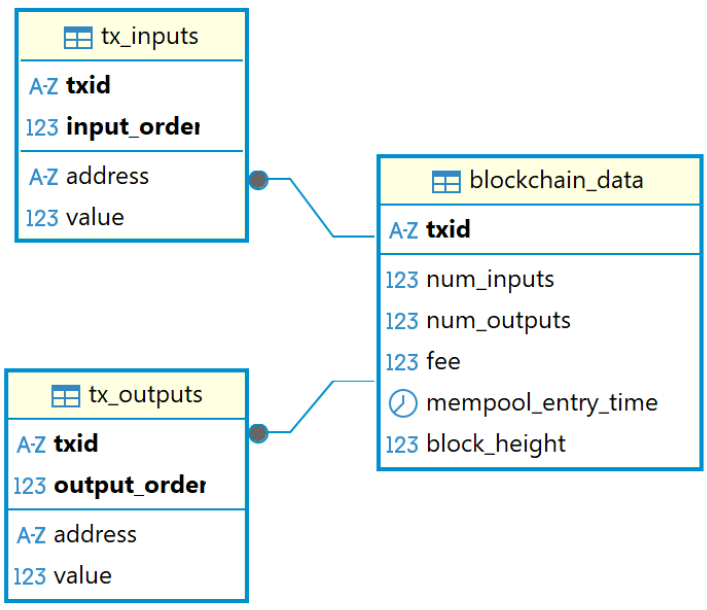
Obr. 23: Štruktúra MySQL tabuľky s údajmi extrahovanými z fóra Bitcointalk

2.2 Krok 2 - validácia adries a zber dát o transakciách

V predchádzajúcom kroku sme získali súbor bitcoinových adries používateľov s uvedenou geografickou lokalitou. V tomto kroku sme pre tieto adresy zozbierali údaje o uskutočnených bitcoinových transakciách. Spracované boli výhradne adresy, ktoré obsahovali vyplnenú bitcoinovú adresu a zároveň automaticky odfiltrovali používateľské profily s neplatným formátom bitcoinovej adresy alebo adresy bez zaznamenanej aktivity.

Transakčné údaje boli získané prostredníctvom blockchainového prehliadača [4], ktorý poskytuje kompletnú databázu blockchainu a umožňuje k nej programový prístup pomocou aplikačného rozhrania (API), bez potreby prevádzkovania vlastného bitcoinového uzla. Prostredníctvom tohto API dokážeme komunikovať s databázou webstránky pomocou určitých vopred definovaných príkazov. V rámci tejto práce bol využívaný najmä príkaz `https://blockchain.info/rawaddr/$bitcoin_address`, kde $\$bitcoin_address$ predstavuje konkrétnu bitcoinovú adresu. Výstupom tohto dotazu je zoznam všetkých prijatých a odoslaných transakcií z danej adresy.

Transakčné dáta sú takmer identické ako sme opísali v časti 1.2.3. Popísaný príklad výstupu je v Príloha A.



Obr. 24: Návrh databázovej schémy na zbieranie transakčných dát

Dáta získané prostredníctvom API sme ukladali do MySQL databázy podobne ako údaje o používateľoch z diskusného fóra Bitcointalk. Databázová schéma pozostáva z troch tabuliek, ktoré sú zobrazené na Obr. 24. Tabuľka **blockchain_data** obsahuje základné informácie o transakciách ako:

- **txid**
- **num_inputs:** Počet vstupov (počet spotrebovaných UTXO).
- **num_outputs:**: Počet výstupov (počet novovytvorených UTXO).
- **fee:** Transakčný poplatok.
- **mempool_entry_time:** Čas vyjadrený v lokálnom časovom pásme (UTC +1:00), od bloku 725 172[2] tento čas vyjadruje kedy bola transakcia prijatá do memory poolu, pri starších blokoch ide o čas schválenia bloku.
- **block_height:** Poradové číslo bloku, v ktorom bola transakcia schválená.

Tabuľka **tx_inputs** obsahuje informácie o všetkých vstupoch do transakcie a obsahuje nasledovné atribúty:

- **txid:** Referencia na príslušnú transakciu.
- **input_order:** Poradové číslo vstupu v transakcii.
- **address:** Zdrojová adresa, z ktorej sa čerpajú UTXO.
- **value:** Hodnota vstupu v bitcoinoch.

Tabuľka **tx_outputs** eviduje informácie o výstupoch transakcie, teda novovytvorených UTXO, a obsahuje tieto atribúty:

- **txid:** Referencia na príslušnú transakciu.
- **output_order:** Poradové číslo výstupu v rámci transakcie.
- **address:** Cieľová adresa, na ktorej sa vytvára nové UTXO.

[2]Blok schválený 27.2.2022 21:09:19

- **value:** Hodnota nového UTXO v bitcoinoch.

Výsledkom tohto kroku bola komplexná databáza transakčnej histórie pre všetky adresy s priradenou geografickou lokalitou. Celkovo bolo identifikovaných 11 800 adries s nenulovým počtom transakcií.

2.3 Krok 3 - priraďovanie časových zón a validácia lokality

V tomto kroku boli textovým reťazcom reprezentujúcim lokalitu priradené príslušné časové pásma. Hoci dostupné knižnice programovacieho jazyka Python umožňujú automatizované mapovanie geografických názvov na časové pásma, pri neštandardných alebo nejednoznačných textových zápisoch lokalít môže byť výsledok nepresný alebo neurčiteľný. Z tohto dôvodu bolo potrebné všetky lokalitné reťazce manuálne skontrolovať a overiť správnosť priradených časových pásiem. Po ukončení tejto validácie zostalo 9 490 bitcoinových adries, pri ktorých bolo možné jednoznačne určiť časové pásmo na základe uvedenej lokality.

2.4 Krok 4 - vylúčenie adries patriacich burzám

V tejto fáze vychádzame z predpokladu, že pre účely predikcie regiónov je potrebné analyzovať výlučne odchádzajúce transakcie z adries používateľov. Prichádzajúce transakcie totiž nie sú priamo viazané na časové pásmo prijímateľa, nakoľko ich realizácia nevyžaduje jeho aktívnu účasť. Pre odosielateľa je postačujúca znalosť bitcoinovej adresy prijímateľa.

Tento metodický prístup však naráža na problém prítomnosti depozitných adries búrz v počiatočnom súbore dát. Ako sme ukázali v časti 1.2.6 používatelia pri týchto platformách nedisponujú spravovaním kľúčov, a preto transakcie odchádzajúce z ich depozitnej adresy nemusia odzrkadľovať ich dennú aktivitu. Preto sme ich pre účely tejto analýzy vylúčili.

Na identifikáciu adries prislúchajúcich k burzám sme využili rozhranie API platformy Arkham Intelligence. Ide o pokročilú blockchainovú analytickú platformu, ktorá sa špecializuje na priraďovanie on-chain entít k reálnym subjektom, ako sú napríklad burzy, investičné fondy, gamblingové platformy alebo známi jednotlivci.

Platforma obsahuje viac než 300 miliónov označení a viac než 150 000 identifikovaných entít, čo z nej robí vysoko spoľahlivý zdroj pre identifikovanie búrz.

API príkaz, ktorý sme v tomto prípade využívali mal nasledovný tvar: https://api.arkm.com/intelligence/address/$, kde symbol $ predstavuje konkrétnu bitcoinovú adresu. Výstupom dotazu bolo označenie adresy ako patriacej ku konkrétnej burze, coinjoinovej službe, minerovi alebo iného označenia zavedeného Arkham platformou, alebo adresy bez známeho priradenia. Pre ďalšiu analýzu boli ponechané iba adresy bez označenia alebo patriace minerom. Po tomto kroku nám zostalo 5 293 adries. Je potrebné poznamenať, že hoci Arkham Intelligence predstavuje robustný nástroj, nemusí pokrývať všetky existujúce burzy či mixovacie služby, a preto nemožno úplne vylúčiť marginálnu prítomnosť inštitucionálnych adries v konečnej vzorke.

2.5 Krok 5 - heuristické zhlukovanie adries

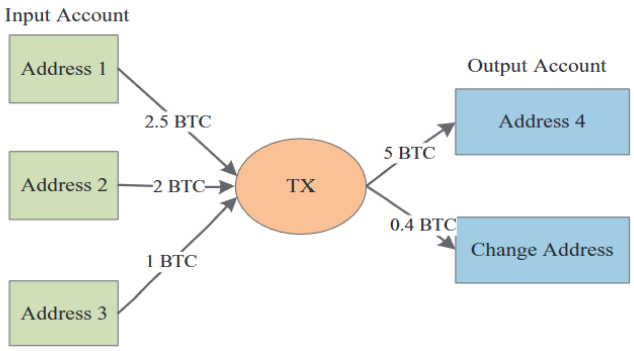
Pre používateľov Bitcoinu je bežné využívanie viacerých bitcoinových adries pre zachovanie anonymity. V kapitole 1 sme naznačili, že väčšina moderných peňaženiek automaticky generuje nové adresy pre každú transakciu. Navyše sa pri každej transakcii vytvára nová adresa pre zvyšok (change address), na ktorú je odoslaná nevyužitá časť vstupných UTXO. V dôsledku toho môže jediný používateľ v priebehu času používať veľké množstvo rôznych adries.

Heuristické zhlukovanie je práve proces, ktorým možno priradiť tieto adresy k jednej entite, respektíve používateľovi. V kontexte tejto práce je prínos heuristického zhlukovania dvojaký: po prvé, rozširuje počet adries, ktoré možno priradiť k jednotlivým používateľom (profilom z diskusného fóra) z nášho datasetu. Po druhé, umožňuje identifikovať adresy patriace burzám alebo iným inštitúciám na základe tvorby extrémne veľkých zhlukov adries.

Pri návrhu algoritmu heuristického zhlukovania sme vychádzali najmä z [11]. Z tejto práce sme prevzali „metódu heuristického zhlukovania založeného na spoločných vstupoch“ (v článku označenú ako H1) a „metódu heuristického zhlukovania založeného na detekcii adresy pre zvyšok“ (v článku označenú ako H2). V ďalšom texte používame rovnaké označenie.

Metóda H1:

Bitcoinové transakcie často čerpajú vstupy (UTXO) z viacerých adries súčasne. Na vytvorenie takejto transakcie je nevyhnutné vlastniť súkromné kľúče ku všetkým použitým vstupným adresám, keďže každý vstup musí byť digitálne podpísaný. Z tohto faktu vyplýva, že všetky adresy vystupujúce ako vstupy v tej istej transakcii, patria a sú kontrolované jedinou entitou. Príklad takejto situácie je zobrazený na Obr. 25, kde transakcia spotrebúva 2.5 BTC z adresy 1, 2 BTC z adresy 2 a 1 BTC z adresy 3. Podľa metódy H1 možno tieto tri adresy priradiť jednej entite. Rovnaký postup bol iteratívne aplikovaný na naše dáta, pričom pre každú analyzovanú transakciu boli jej vstupné adresy zlúčené do jednej entity.



Obr. 25: Schéma vstupov a výstupov transakcie [11]

Metóda H2:

Táto metóda je založená na identifikácii adresy pre zvyšok. Ako je znázornené na Obr. 25, transakcia obsahuje 2 výstupy: 5 BTC sa posiela na adresu 4 a 0.4 BTC je odoslaných naspäť odosielateľovi na adresu pre zvyšok. Táto situácia vzniká preto, že súčet vstupných UTXO presahuje sumu určenú na odoslanie, a preto sa zvyšná hodnota musí vrátiť späť odosielateľovi, tak ako sme ukázali v časti 1.2.3.

Keďže bitcoinový protokol explicitne neoznačuje, ktorý z výstupov predstavuje adresu pre zvyšok, neexistuje absolútne spoľahlivá metóda jej detekcie. V tejto práci sme implementovali rozhodovací algoritmus navrhnutý a overený v [11].

Myšlienkový proces algoritmu je nasledovný. Predpokladáme, že sledujeme určitú bitcoinovú transakciu s cieľom objavenia adresy pre zvyšok. Najskôr sa overí, či niektorá z výstupných adries je totožná s niektorou zo vstupných adries. Ak takáto

adresa existuje, transakcia je označená ako obsahujúca vlastnú adresu pre zvyšok (self-change) a táto adresa je považovaná za adresu pre zvyšok.

Ak taká neexistuje, posudzuje sa počet výstupov. Pri jednom výstupe sa predpokladá, že adresa pre zvyšok neexistuje. Pri dvoch výstupoch sa overuje, či jedna z výstupných adries vystupuje v celom blockchaine ako výstup iba raz a zároveň či hodnota jej výstupu má aspoň o tri desatinné miesta viac než druhý výstup. Ak je táto podmienka splnená, daná adresa je označená ako adresa pre zvyšok. Pri viac než dvoch výstupoch sa najskôr identifikujú adresy, ktoré sa v celom blockchaine vyskytujú ako výstup iba raz. Ak medzi výstupmi existuje presne jedna takáto adresa, je označená ako adresa pre zvyšok. Detailne popísaný vývojový diagram uvádzame v Príloha B.

Kombináciou oboch heuristických metód dochádza k rozšíreniu množiny adries k jednotlivým entitám a tým aj k zvýšeniu počtu transakcií. Zároveň tento postup odhalil potenciálne adresy patriace burzám alebo coinjoinovým službám. Ak zhluk adries priradený jednej entite dosahuje neštandardne veľký rozsah, predpokladali sme, že ide o infraštruktúru nejakej inštitúcie. Takéto anomálne zhluky boli manuálne vylúčené. Okrem toho boli všetky nové adresy získané heuristickým zhlukovaním opätovne overené prostredníctvom API platformy Arkham Intelligence (Krok 4). Celkový počet entít po tomto kroku bol 5 063.

Vývoj počtu entít sme znázornili v tabuľke 1.

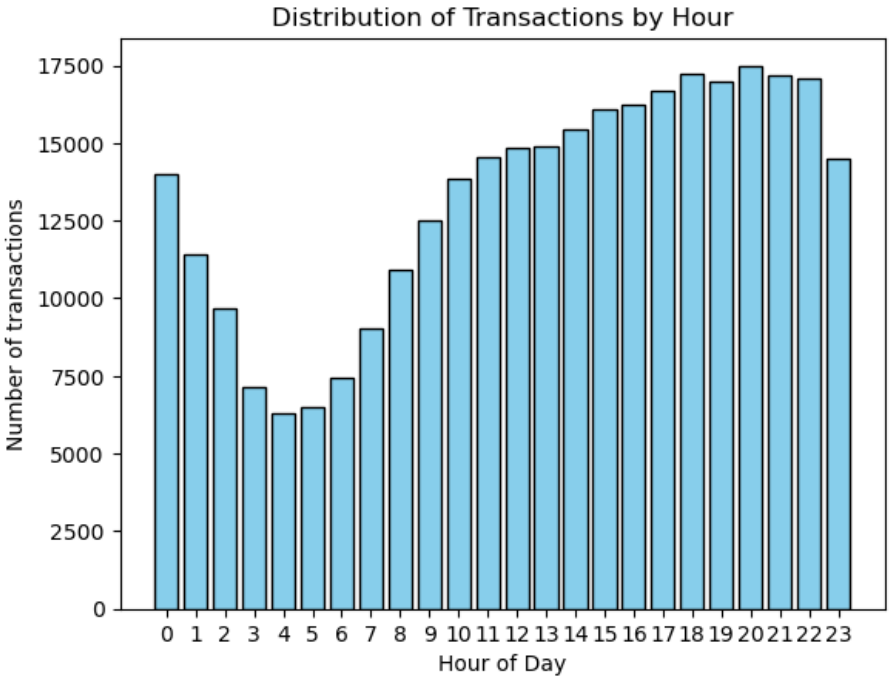
Opis metódy	**Zostávajúci počet entít**
Registrovaní používatelia na Bitcointalk	3 700 000
Používatelia s vyplnenou BTC adresou	129 058
Používatelia s vyplnenou BTC adresou a lokalitou	77 561
Overenie nenulového počtu transakcií	11 800
Overenie existencie lokality a priradenie časového pásma	9 490
Vylúčenie zmenárni	5 292
Heuristické zhlukovanie	5 063

Tabuľka 1: Prehľadná tabuľka vývoja počtu entít

2.6 Krok 6 - transakcie a regionálne zhlukovanie časových pásiem

V záverečnom kroku zberu dát boli k jednotlivým entitám extrahované transakčné údaje z vytvorenej databázy a následne im bolo priradené príslušné časové pásmo. Primárnym cieľom tejto fázy bolo identifikovať, či v časovej distribúcii transakčnej aktivity existuje pozorovateľný globálny vzor.

Na Obrázku 2.5 je zobrazený histogram agregovanej aktivity naprieč všetkými transakciami a časovými pásmami, pričom referenčný čas je vyjadrený v štandarde UTC +0:00. Z grafu je evidentný výrazný pokles aktivity v intervale medzi 2:00 a 7:00, kedy je objem transakcií viac než dvojnásobne nižší v porovnaní s dennou prevádzkou. Takéto okno časovej neaktivity bolo prítomné aj pri jednotlivých časových zónach. Očakávame, že v závislosti od polohy tohto neaktívneho okna, je možné rozlíšiť pôvod používateľov. Toto empirické zistenie predstavuje kľúčové východisko pre návrh predikčného klasifikačného modelu.



Obr. 26: Agregovaný histogram aktivity transformovaný na časovú zónu UTC +0:00

Uvedené pozorovanie bolo prvotne analyzované na úrovni izolovaných transakcií. Napriek tomu, že agregovaný histogram vykazoval zreteľný trend, použitie jed-

notlivých transakcií ako priamych vstupov pre model neposkytovalo signifikantné výsledky. Izolovaná transakcia totiž nenesie dostatočnú informačnú hodnotu o celkovom profile aktivity používateľa. Z tohto dôvodu bol zvolený prístup trénovania modelov na dátach agregovaných na úroveň entít.

Pre každú entitu boli jej transakcie rozdelené do 24 časových segmentov (0-23), kde každý segment reprezentuje jednu hodinu dňa. Následne bola vykonaná normalizácia týchto hodnôt za účelom získania relatívneho podielu transakcií v jednotlivých hodinách. Do výpočtu bola implementovaná metóda aditívneho vyhladzovania. Ku každej hodine bola umelo pripočítaná konštanta 1, aby sa pri entitách s nízkym celkovým počtom transakcií utlmil vplyv ojedinelých aktívnych hodín, ktoré nemusia štatisticky spoľahlivo odrážať skutočný behaviorálny vzor používateľa. Relatívnu frekvenciu transakcií pre konkrétnu hodinu i (napríklad pre interval $[0:00, 1:00)$, kde $i = 0$) je možné formálne vyjadriť vzťahom:

$$\text{Relatívna aktivita}_i = \frac{\#Tr_i + 1}{\sum_{j=0}^{23} \#Tr_j + 24},$$

kde $\#Tr_i$ vyjadruje počet transakcií vykonaných v časovom intervale $[i, i+1)$. Tento normalizačný postup bol plošne aplikovaný na všetky časové segmenty. Predpokladom tejto transformácie je, že agregovaný a vyhladený časový profil dokáže zachytiť trend aktivity výrazne presnejšie než analýza surových transakcií.

Distribúcia entít naprieč jednotlivými časovými pásmami je kvantifikovaná v Tabuľke 2. Z tabuľky vyplýva, že majoritná časť používateľov (71,12 %) pochádza zo šiestich dominantných časových pásiem: -5, 1, 2, 5.5, 7 a 8. Zvyšné časové zóny vykazujú len marginálne zastúpenie, z dôvodu čoho bola pre zvýšenie robustnosti a výpočtovej efektívnosti modelu navrhnutá ich agregácia.

Histogram časovej aktivity navyše indikuje, že primárnym diskriminačným znakom pre identifikáciu časového pásma entity bude detekcia okna neaktivity (spánku). Vzhľadom na to, že priemerné okno neaktivity pokrýva interval viacerých hodín a pri používateľoch z rovnakej časovej zóny môže byť mierne odlišné, predpokladáme zníženú schopnosť klasifikátora exaktne rozlišovať medzi úzko susediacimi časovými pásmami. Preto sme zvolili stratégiu zhlukovania do štyroch geografických regiónov: Amerika (časové zóny $[-8, -3]$), Európa s Afrikou (zóny $[0, 3]$), Centrálna Ázia (zóny $[3.5, 6.5]$) a Východná Ázia s Pacifikom (zóny $[7, 13]$ a $[-10, -9]$). Výsledné

rozloženie používateľov a transakcií po tejto agregácii mapuje Tabuľka 3. Môžeme si všimnúť rozdiely medzi mediánom a priemerom počtu transakcií na entitu, medián naznačuje existenciu veľkého množstva entít s nízkym počtom transakcií. Tieto entity nedokážu dostatočne dobre reprezentovať rozdelenie aktivity počas dňa, preto boli neskôr vylúčené. Na druhú stranu vyššia hodnota priemeru potvrdzuje existenciu entít s veľkým počtom transakcií. Tieto entity môžu predstavovať individuality s vysokou transakčnou aktivitou alebo môžu predstavovať aj iný nie individuálny subjekt. Po manuálnej kontrole týchto entít sme sa ich rozhodli ponechať pretože nevykazovali známky búrz alebo iných komerčných služieb.

Čas. pásmo	**Počet entít**	**Počet entít %**	**Čas. pásmo**	**Počet entít**	**Počet entít %**
-10	7	0.15	4.5	2	0.04
-8	152	3.25	5	87	1.86
-7	53	1.13	5.5	589	12.57
-6	154	3.29	5.75	15	0.32
-5	654	13.96	6	65	1.39
-4	37	0.79	6.5	50	1.07
-3	118	2.52	7	506	10.8
0	272	5.81	8	505	10.78
1	749	15.99	9	38	0.81
2	329	7.02	10	8	0.17
3	154	3.29	10.5	5	0.11
3.5	16	0.34	11	76	1.62
4	25	0.53	13	18	0.38

Tabuľka 2: Distribúcia počtu entít podľa jednotlivých časových pásiem

Región	Počet entít	Počet entít %	Počet transakcií	Priemerný počet transakcií na entitu	Mediánový počet transakcií na entitu
Amerika	1 168	24.94	64 669	55.37	7
Euroafrika	1 504	32.11	126 774	84.29	12
Centrálna Ázia	849	18.13	56 867	66.98	15
Východná Ázia	1 163	24.83	69 533	59.79	7

Tabuľka 3: Prehľad počtu entít a objemu transakcií po agregácii do regiónov

3 Klasifikačné modely

V predchádzajúcej kapitole sme pripravili dátovú sadu obsahujúcu 24 numerických premenných a jednu cieľovú premennú so štyrmi triedami. Našou úlohou je na základe týchto dát predikovať regióny, z ktorých jednotliví používatelia siete Bitcoin pochádzajú. Na tento účel aplikujeme vybrané klasifikačné algoritmy strojového učenia.

V prvej časti tejto kapitoly predstavíme matematický princíp logistickej regresie, ktorú sme zvolili ako základný (referenčný) model. Ako sme však ukázali v analýze dát, veľká časť transakčnej aktivity používateľov neposkytuje dostatok informácií na jednoznačné a deterministické určenie ich lokality. Z tohto dôvodu potrebujeme modely, ktoré na výstupe neposkytujú iba zaradenie do triedy, ale dokážu kvantifikovať aj pravdepodobnosť tohto zaradenia (mieru istoty). Hoci logistická regresia túto požiadavku spĺňa, pre zachytenie komplexných nelineárnych závislostí medzi 24 premennými sme sa v tejto práci rozhodli zamerať predovšetkým na metódy založené na rozhodovacích stromoch. Zameriame sa najmä na XGBoost, ktorého matematické koncepty si predstavíme v tejto kapitole. Nakoniec uvedieme a popíšeme význam kritérií kvality, pomocou ktorých budeme analyzovať správanie a predikčnú schopnosť klasifikačných modelov.

Najskôr si uveďme základné značenie, ktoré budeme používať v rámci celej tejto kapitoly. Nech $X \in \mathbb{R}^{N \times P}$ je matica vstupných, trénovacích dát predstavujúca premenné $p = 1, 2, \ldots, P$ pre jednotlivé pozorovania $n = 1, 2, \ldots, N$:

$$X = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1P} \\ x_{21} & x_{22} & \ldots & x_{2P} \\ \vdots & \vdots & \ddots & \vdots \\ x_{N1} & x_{N2} & \ldots & x_{NP} \end{pmatrix}$$

Cieľovú premennú (target variable) označíme $Y \in \mathbb{R}^N$, kde vo všeobecnosti uvažujeme, že pre každé pozorovanie nadobúda y_i jednu z diskrétnych K tried. Hoci sú väčšinou tieto triedy kategorické, pre potreby matematického modelovania ich štandardne transformujeme na celočíselné indexy. Formálne teda priestor možných hodnôt pre každé pozorovanie definujeme ako $y_i \in \{1, 2, \ldots, K\}$.

3.1 Logistická regresia

V tejto časti si predstavíme teoretické základy logistickej regresie, pričom budeme vychádzať z odbornej literatúry [10] a [12]. Keďže princíp tohto algoritmu bol pôvodne odvodený pre riešenie úloh binárnej klasifikácie, začneme s binárnou klasifikáciou, pre ktorú definujeme cieľovú premennú ako $y_i \in \{0, 1\}$. Následne rozšírime teóriu aj o K-klasifikáciu, kde sa vrátime aj k pôvodnému definovaniu y_i.

Logistická regresia nemodeluje príslušnosť k triede priamo, ale predikuje pravdepodobnosť, že dané pozorovanie patrí do pozitívnej kategórie ($y_i = 1$). Od modelu preto vyžadujeme, aby na výstupe generoval výlučne hodnoty z uzavretého intervalu $[0, 1]$. To predstavuje zásadný rozdiel oproti štandardnej lineárnej regresii. Lineárny model má tvar $\hat{y}(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_P x_P$, kde x predstavuje vektor P vstupných premenných a vektor $\beta \in \mathbb{R}^{P+1}$ sú hľadané parametre modelu. Takáto lineárna kombinácia však môže vo všeobecnosti nadobúdať ľubovoľné reálne hodnoty ($\hat{y}(x) \in \mathbb{R}$).

Pre transformáciu výstupu lineárneho modelu do požadovaného pravdepodobnostného intervalu $[0, 1]$ je použitá logistická funkcia:

$$p(x, \beta) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_P x_P)}}$$

Takto získaná hodnota $p(x, \beta)$ reprezentuje pravdepodobnosť zaradenia pozorovania do pozitívnej triedy, čo možno formálne zapísať ako podmienenú pravdepodobnosť $P(Y = 1 \mid X = x_i) = p(x_i, \beta)$. Algebraickou úpravou tohto vzťahu vieme vyjadriť šance (odds):

$$\frac{p(x_i, \beta)}{1 - p(x_i, \beta)} = e^{\beta_0 + \beta_1 x_{i1} + \cdots + \beta_P x_{iP}}$$

Tento pomer (odds) poskytuje iný, často intuitívnejší pohľad na výstup modelu, než samotná pravdepodobnosť. Matematicky vyjadruje, koľkokrát je pravdepodobnejšie, že skúmaný jav nastane, v porovnaní s tým, že nenastane. Ak by napríklad model predikoval pre konkrétne pozorovanie hodnotu šancí rovnej 0.2 (teda pomer 1 : 5), znamenalo by to, že pravdepodobnosť zaradenia tohto pozorovania do pozitívnej triedy je päťkrát menšia ako pravdepodobnosť jeho zaradenia do negatívnej triedy. Naopak, ak by bol pomer šancí napríklad 4, zaradenie do pozitívnej triedy by bolo štyrikrát pravdepodobnejšie.

Parametre β odhadujeme v prípade logistickej regresie metódou maximálnej vierohodnosti (Maximum Likelihood Estimation, MLE). Základná intuícia spočíva v hľadaní takých parametrov β, aby predikovaná pravdepodobnosť bola čo najbližšie k reálnym (y_i), ktoré vidíme v trénovacej sade. Pre konkrétne pozorovanie definujeme rozdelenie pravdepodobnosti nasledovne:

$$P(Y = y_i|X = x_i) = p(x_i, \beta)^{y_i} * (1 - p(x_i, \beta))^{1-y_i}$$

Tento vzťah zaručuje, že ak je skutočná trieda $y_i = 1$, pravdepodobnosť klasifikácie je rovná $p(x_i, \beta)$, a ak je $y_i = 0$, pravdepodobnosť je $1 - p(x_i, \beta)$.

Za predpokladu, že pozorovania sú nezávislé a rovnako rozdelené (i.i.d.), môžeme pravdepodobnosť správnej klasifikácie pre celú trénovaciu množinu vyjadriť ako súčin jednotlivých pravdepodobností, čím získame funkciu vierohodnosti:

$$L(x, \beta) = \prod_{i=1}^{N} P(Y = y_i|X = x_i)$$

Cieľom metódy MLE je nájsť $\beta^* = \underset{\beta}{argmax} L(x, \beta)$. Keďže logaritmus je rastúca funkcia, môžeme namiesto priamej vierohodnosti maximalizovať jej logaritmus (log-likelihood). Z numerických dôvodov (prechod na minimalizáciu) definujeme stratovú funkciu ako záporný logaritmus vierohodnosti:

$$\beta^* = \underset{\beta}{argmin} \left[-\sum_{i=1}^{N} (y_i \log(p(x_i, \beta)) + (1 - y_i) \log(1 - p(x_i, \beta))) \right]$$

Takýto zápis je identický s tým, čo sa nachádza v knižnici scikit-learn v pythone [16]. Riešením tejto úlohy sa v tejto práci nebudeme zaoberať, jeden z možných prístupov ako ju vyriešiť bol predstavený v [10], kde bolo riešenie nájdené pomocou metód voľnej optimalizácie, konkrétne Newtonovou metódou. Po nájdení optimálnych parametrov β^* získame všeobecný model, schopný predikcie pravdepodobnosti aj pre nové dáta, nenachádzajúce sa v trénovacej sade.

Pri rozšírení logistickej regresie na problém klasifikácie do K tried modelujeme relatívny pomer pravdepodobností medzi jednotlivými triedami a jednou zafixovanou referenčnou triedou. Ak za referenčnú zvolíme poslednú triedu K, vytvoríme $K - 1$ samostatných lineárnych modelov pre logaritmus pomeru šancí. Celú túto

sústavu môžeme explicitne rozpísať nasledovne:

$$\begin{aligned}
\frac{P(Y=1|X=x_i)}{P(Y=K|X=x_i)} &= e^{\beta_{10}+\beta_{11}x_{i1}+\cdots+\beta_{1P}x_{iP}} \\
\frac{P(Y=2|X=x_i)}{P(Y=K|X=x_i)} &= e^{\beta_{20}+\beta_{21}x_{i1}+\cdots+\beta_{2P}x_{iP}} \\
&\vdots \\
\frac{P(Y=K-1|X=x_i)}{P(Y=K|X=x_i)} &= e^{\beta_{K-1,0}+\beta_{K-1,1}x_{i1}+\cdots+\beta_{K-1,P}x_{iP}}
\end{aligned}$$

Tieto zlomky vyjadrujú, koľkokrát je pravdepodobnejšie, že pozorovanie patrí do konkrétnej triedy k v porovnaní s referenčnou triedou K.

Pre ďalšie sprehľadnenie matematického zápisu si zadefinujme túto lineárnu kombináciu premenných pre k-tu triedu v tvare vektorového súčinu: $\beta_k^T x_i = \beta_{k0}+\beta_{k1}x_{i1}+\cdots+\beta_{kP}x_{iP}$. Keďže vieme, že súčet pravdepodobností všetkých tried musí byť rovný 1 (teda $\sum_{l=1}^{K} P(Y = l \mid X = x_i) = 1$), algebraickými úpravami predchádzajúcej sústavy rovníc dokážeme explicitne vyjadriť pravdepodobnosť pre referenčnú triedu K a pre zvyšné triedy k:

$$P(Y = K \mid X = x_i) = \frac{1}{1+\sum_{l=1}^{K-1} e^{\beta_l^T x_i}}$$

$$P(Y = k \mid X = x_i) = \frac{e^{\beta_k^T x_i}}{1+\sum_{l=1}^{K-1} e^{\beta_l^T x_i}} \quad \text{pre } k = 1, 2, \ldots, K-1$$

Hoci je tento prístup s pevnou referenčnou triedou štatisticky korektný a postačujúci, do výpočtov vnáša určitú asymetriu. V modernom strojovom učení sa preto častejšie využíva symetrickejšia formulácia, pri ktorej nevyberáme referenčnú triedu. Namiesto toho priradíme plnohodnotný vektor parametrov β každej z K tried. Táto transformácia je v odbornej literatúre známa ako Softmax funkcia:

$$p_k(x, \beta) = P(Y = k \mid X = x) = \frac{e^{\beta_k^T x}}{\sum_{l=1}^{K} e^{\beta_l^T x}}$$

V tomto prípade optimalizačná úloha pre hľadanie optimálnych parametrov β bude vyzerať nasledovne:

$$\beta^* = \underset{\{\beta_k\}_{k=1}^K}{argmin} \left[-\sum_{i=1}^{N} \sum_{k=1}^{K} I(y_i = k) log(p_k(x, \beta)) \right],$$

kde funkcia $I(y_i = k)$ nadobúda hodnotu 1, ak je skutočná trieda pozorovania k, a v opačnom prípade je rovná 0. Výsledkom tejto optimalizácie je všeobecný model, schopný predikovať pravdepodobnosti príslušnosti do jednotlivých tried pre nové dáta.

3.2 XGBoost

V tejto časti si predstavíme matematickú myšlienku použitia metódy XGBoost (Extreme Gradient Boosting), ktorý je postavený na boostingu. Boosting patrí medzi metódy súborového učenia (ensemble methods), ktoré fungujú na princípe kombinovania viacerých jednoduchších bázických modelov do jedného potenciálne veľmi kvalitného modelu. Medzi tieto metódy patria aj random forests a bagging. Pri výbere jednoduchých bázických modelov pre tieto metódy môže byť aplikovateľné široké spektrum štatistických metód. My sa v tejto práci zameriame na použitie v kombinácii s rozhodovacími stromami.

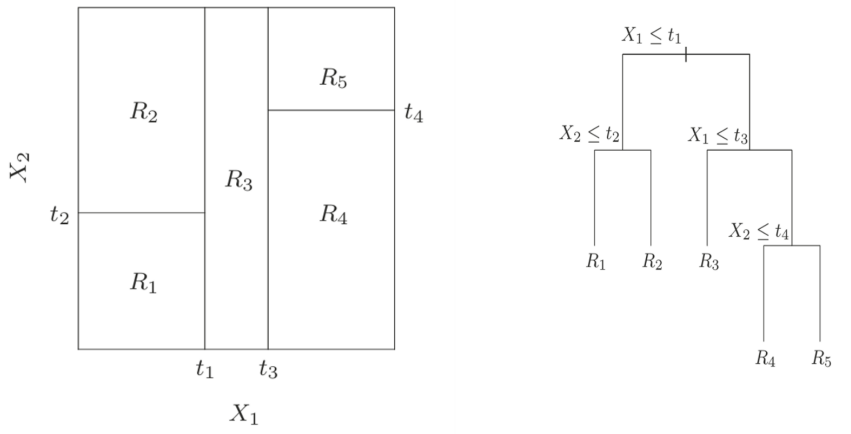
V tejto podkapitole si postupne predstavíme matematické koncepty rozhodovacích stromov, kde sme čerpali najmä zo zdrojov [10] a [12]. Značenie použité pri definovaní pojmov ako aj zastavovacie kritériá delenia sme prebrali zo zdrojového kódu knižnice scikit-learn v Pythone [16]. V prípade boostingu sme čerpali najmä zo zdrojov [10] a [12]. Ďalej si predstavíme v tejto časti Gradient Boosting, ktorého vysvetlenie je podstatné pre pochopenie fungovania a odvodenie metódy XGBoost. Pri týchto dvoch metódach sme čerpali zo zdrojov [10], [8] a [6].

3.2.1 Rozhodovacie stromy

Rozhodovacie stromy sú založené na princípe rozdelenia p-rozmerného priestoru premenných na J disjunktných oblastí $R_1, R_2 \dots R_J$. Každému pozorovaniu, ktoré padne do oblasti R_j, priradí model rovnakú hodnotu. Na Obr. 27 vľavo je ilustrované takéto rozdelenie dvojrozmerného priestoru premenných na 5 disjunktných oblastí. Vytvorenie takéhoto rozdelenia sa realizuje algoritmicky. V ilustračnom príklade sa najskôr rozdelil celý priestor podľa pravidla $X_1 = t_1$. Následne sa región $X_1 \leq t_1$ rozdelil v bode $X_2 = t_2$ a región $X_1 > t_1$ v $X_1 = t_3$. Nakoniec región $X_1 > t_3$ sa rozdelil v bode $X_2 = t_4$.

Hierarchickú štruktúru takéhoto modelu je možné vizualizovať vo forme binárneho stromu (Obr. 27 vpravo). V kontexte teórie grafov reprezentuje aplikácia každého rozhodovacieho pravidla ($X_i \leq t$) uzol (node) stromu. Počiatočný uzol, reprezentujúci celý nerozdelený priestor premenných, sa označuje ako koreň (root). Koncové body grafu, ktoré vzniknú po aplikácii všetkých predchádzajúcich kritérií,

nazývame terminálnymi uzlami alebo listami. Rozdelenie priestoru je plne zachytené štruktúrou stromu, pričom každý list reprezentuje jednu vytvorenú disjunktnú oblasť.



Obr. 27: Ilustrácia rozdelenia dvojrozmerného priestoru premenných (vľavo) a jeho reprezentácia pomocou binárneho rozhodovacieho stromu (vpravo). (Zdroj: [12])

Rozhodovací strom môže byť teda formálne vyjadrený ako:

$$T(x, \omega) = \sum_{j=1}^{J} c_j I(x \in R_j)$$

$$I(x \in R_j) = \begin{cases} 1 & \text{ak } x \in R_j \\ 0 & \text{ak } x \notin R_j \end{cases},$$

s parametrami $\omega = \{R_j, c_j\}_1^J$. Hoci je základný koncept rozhodovacích stromov priamočiary, kľúčovým problémom ostáva hľadanie optimálnych parametrov c_j a disjunktných oblastí R_j.

Určenie parametrov c_j pri už definovaných disjunktných oblastiach je spravidla priamočiare. V prípade regresie sa hodnota c_j vypočíta ako aritmetický priemer cieľových premenných y_i patriacich do danej oblasti R_j. V prípade klasifikácie je c_j stanovené ako najčastejšie vyskytujúca sa trieda v R_j.

Hľadanie optimálnych disjunktných oblastí predstavuje komplexnejší problém. Zadefinujme si najskôr formálne označenie. Dáta v uzle v budeme označovať ako Q_v, pričom n_v predstavuje počet vzoriek v tomto uzle. Pre každé potencionálne rozdelenie $\theta = (p, t_v)$, pozostávajúce z rozdeľovacej premennej p a rozdeľovacej hodnoty

t_v, definujme polroviny $Q_v^{right}(\theta)$ a $Q_v^{left}(\theta)$ nasledovne:

$$Q_v^{left}(p, t_v) = \{(x, y)|x_p \leq t_v\} \; a \; Q_v^{right}(p, t_v) = \{(x, y)|x_p > t_v\}$$

Kvalitu rozdelenia v uzle v vieme numericky kvantifikovať prostredníctvom funkcie nečistoty uzla (impurity function) H. Pre výber optimálnych parametrov delenia riešime minimalizačnú úlohu:

$$\theta^* = \underset{\theta}{argmin} \;\; \frac{n_v^{left}}{n_v} H(Q_v^{left}(\theta)) + \frac{n_v^{right}}{n_v} H(Q_v^{right}(\theta)) \tag{1}$$

V prípade regresie sa ako funkcia nečistoty najčastejšie využíva stredná štvorcová chyba (MSE):

$$H(Q_v) = \frac{1}{n_v} \sum_{x_i \in Q_v} (y_i - \overline{y}_v)^2 \quad \text{, kde} \quad \overline{y}_v = \frac{1}{n_v} \sum_{x_i \in Q_v} y_i$$

V prípade klasifikácie, kde možné klasifikačné triedy sú $1, 2, \ldots K$, označme p_{vk} ako proporciu triedy k v uzle v:

$$p_{vk} = \frac{1}{n_v} \sum_{x_i \in Q_v} I(y_i = k)$$

$$I(y_i = k) = \begin{cases} 1 & \text{ak } y_i = k \\ 0 & \text{ak } y_i \neq k \end{cases}$$

Potom pre klasifikáciu rozoznávame nasledujúce funkcie nečistoty:

Giniho index (Gini index): $H(Q_v) = \sum_{k=1}^{K} p_{vk}(1 - p_{vk})$

Krížová entropia (Cross-entropy): $H(Q_v) = -\sum_{k=1}^{K} p_{vk} \log(p_{vk})$

Pre efektívne hľadanie riešenia úlohy 1 sa využíva napríklad greedy algoritmus CART, ktorý pozostáva z nasledovných krokov:

1. **Inicializácia:** Algoritmus začína s celým priestorom premenných a všetkými pozorovaniami. Pre každú prípustnú premennú $p = 1, 2, \ldots P$ a t_v sa priestor diskretizuje identifikáciou všetkých kandidátskych bodov rezu t_v. Tieto body sú stanovené ako priemery susedných, vzostupne usporiadaných unikátnych hodnôt danej premennej: $\frac{x_{i,p} + x_{i+1,p}}{2}$

2. **Vyhodnotenie rezov:** Dosadením všetkých prípustných dvojíc (p, t_v) sa pre zvolenú funkciu nečistoty vyhodnotí účelová funkcia definovaná vzťahom 1. Optimálne rozdelenie θ^* je identifikované na základe minimálnej dosiahnuteľnej hodnoty tejto funkcie.

3. **Rekurzia:** Proces sa rekurzívne aplikuje na novovzniknuté podmnožiny dát. Po nájdení optimálneho rozdelenia θ^* v uzle v pokračuje algoritmus v hľadaní optimálneho rozdelenia pre ľavý pod-uzol Q_v^{left} a pravý pod-uzol Q_v^{right} samostatne.

4. **Ukončovacie kritériá**: Rekurzívne delenie sa zastaví pri splnení aspoň jedného z nasledujúcich kritérií:

   (a) **Maximálna hĺbka (max depth):** Definuje maximálny počet hrán (rozhodovacích krokov) na ceste od koreňa k terminálnemu uzlu.

   (b) **Minimálny počet vzoriek na delenie (min samples split):** Definuje minimálny počet pozorovaní v uzle v (n_v) nevyhnutných na to, aby algoritmus vôbec zvažoval ďalšie delenie.

   (c) **Minimálny počet vzoriek v liste (min samples leaf):** Zamedzuje vytvoreniu rezu v prípade, ak by v niektorom z novovzniknutých listov zostal menší počet dát, ako je táto prahová hodnota.

   (d) **Minimálny pokles nečistoty (min impurity decrease):** Uzol sa rozdelí iba za predpokladu, že pokles celkovej nečistoty prekročí stanovenú prahovú hodnotu. Pokles nečistoty je definovaný ako:

$$\frac{n_v}{N}\left(H(Q_v) - \frac{n_v^{left}}{n_v}H(Q_v^{left}) - \frac{n_v^{right}}{n_v}H(Q_v^{right})\right)$$

5. **Finálne priradenie:** Výsledkom procesu je plne vygenerovaný binárny strom s celkovým počtom uzlov V. Terminálne uzly (listy), ktorých počet je $J = V + 1$, následne formujú výsledné disjunktné oblasti $R_1, R_2, \ldots, R_J$.

Je dôležité poznamenať, že klasifikačné rozhodovacie stromy neposkytujú iba „tvrdú“ klasifikáciu (hard classification), teda priame priradenie pozorovania k jednej triede, ale prostredníctvom pravdepodobností p_{vk} v terminálnych uzloch umožňujú

aj „jemnú“ klasifikáciu (soft classification). V tomto režime je ľubovoľnému pozorovaniu x priradený vektor pravdepodobností príslušnosti ku každej z K definovaných tried.

3.2.2 Boosting

Boosting patrí medzi súborové metódy (ensemble methods), ktoré fungujú na princípe kombinovania viacerých jednoduchých a menej presných modelov (v literatúre označovaných aj ako elementárne alebo bázické funkcie) s cieľom vytvorenia jedného presného modelu. Medzi súborové metódy patria napríklad aj metódy bagging a náhodné lesy, ktoré trénujú jednotlivé bázické funkcie paralelne na náhodných podmnožinách pozorovaní alebo premenných. Na druhej strane pri boostingu sa séria jednoduchých modelov trénuje sekvenčne, pričom každá nasledujúca bázická funkcia využíva informáciu z predchádzajúcich krokov a snaží sa priamo korigovať chyby svojich predchodcov.

Výsledkom týchto modelov je aditívny model, v ktorom je konečná predikcia daná súčtom všetkých bázických modelov. Ako takúto elementárnu funkciu si môžeme predstaviť napríklad málo rozkonárený, plytký rozhodovací strom. Hoci vo všeobecnosti môžu rolu bázických funkcií plniť rôzne typy modelov, v tejto práci sa zameriame výlučne na aditívny model využívajúci rozhodovacie stromy. Takýto model možno formálne zapísať nasledovne:

$$f(x) = f_M(x) = \sum_{m=1}^{M} T(x, \omega_m) = \sum_{m=1}^{M} \sum_{j=1}^{J} c_{jm} I(x \in R_{jm}), \tag{2}$$

Takéto modely sa štandardne optimalizujú minimalizáciou zvolenej globálnej stratovej funkcie:

$$\underset{\{\omega_m\}_1^M}{min} \quad \sum_{i=1}^{N} L\left(y_i, \sum_{m=1}^{M} T(x_i, \omega_m)\right) \tag{3}$$

Priame analytické riešenie tohto optimalizačného problému je vo všeobecnosti neuskutočniteľné. V praxi sa preto využíva prístup dopredného po-krokového aditívneho modelovania (Forward Stagewise Additive Modeling), ktorý aproximuje riešenie úlohy 3 sekvenčným pridávaním nových bázických funkcií. V každom kroku sa do modelu pridáva nová bázická funkcia, pre ktorú sa hľadajú optimálne parametre ω_m. Tento algoritmus možno formálne vyjadriť nasledovne:

1. Zvolíme $f_0(x) = 0$

2. Pre $m = 1$ do M:

   (a) Vypočítame

$$\omega_m = \underset{\omega}{argmin} \sum_{i=1}^{N} L\left(y_i, f_{m-1}(x_i) + T(x_i, \omega)\right)$$

   (b) Zvolíme $f_m(x) = f_{m-1}(x) + T(x, \omega_m)$

Podobne ako pri izolovaných rozhodovacích stromoch, aj v každej iterácii aditívneho modelu nastáva optimalizačný problém určenia konštánt c_{jm} a identifikácie disjunktných oblastí R_{jm} prislúchajúcich aktuálnemu m-tému kroku.

Ak sú disjunktné regióny R_{jm} už známe, určenie optimálnych konštánt c_{jm} sa spravidla realizuje analytickým alebo numerickým riešením nasledujúcej minimalizačnej úlohy pre každý región samostatne:

$$c^*_{jm} = \underset{c_jm}{argmin} \sum_{i \in S_{jm}} L(y_i, f_{m-1}(x_i) + c_{jm}) \quad \text{, kde } S_{jm} = \{i | x_i \in R_{jm}\}$$

Samotné hľadanie optimálnych disjunktných podmnožín (topológie stromu) je však podstatne komplikovanejšou úlohou než pri samostatných rozhodovacích stromoch. V určitých špecifických prípadoch sa tento matematický problém zjednodušuje. Napríklad, pri úlohách binárnej klasifikácie s využitím exponenciálnej stratovej funkcie je možné z tohto všeobecného rámca analyticky odvodiť známy algoritmus AdaBoost.

3.2.3 Gradient boosting

Gradient Boosting je ďalšia metóda určená na riešenie dopredného po-krokového aditívneho modelovania (Forward Stagewise Additive Modeling), ktorej cieľom je minimalizovať účelovú funkciu 3.

Pre ľubovoľnú diferencovateľnú stratovú funkciu môže byť odvodené riešenie pomocou numerických metód voľnej optimalizácie. Ak uvážime, že stratová funkcia vyčísľuje odchýlku predikčného modelu od skutočných cieľových premenných, tak naším cieľom je nájsť taký vektor $\overline{f}$, ktorý túto funkciu minimalizuje. Stratovú

funkciu môžeme vyjadriť aj ako: $L(\overline{f}) = \sum_{i=1}^{N} L(y_i, f(x_i))$, kde $f(x)$ reprezentuje výsledok predikcie aditívneho modelu popísaného vo vzorci 2. Podľa zdroja [9] sa na túto úlohu môžeme pozerať ako na viacrozmernú optimalizačnú úlohu na voľný extrém:

$$\begin{aligned} \overline{f}^* = \underset{\overline{f}}{argmin} \quad & L(\overline{f}) \\ & \overline{f} \in \mathbb{R}^N \end{aligned}$$

Parametrom tejto optimalizačnej úlohy je vektor $\overline{f}$, predstavujúci hodnoty predikčnej funkcie aditívneho rozvoja pre každé z N pozorovaní x_i, teda

$$\overline{f} = (f(x_1), f(x_2), \ldots, f(x_N))^T$$

Takéto úlohy možno riešiť iteračnými metódami, ktoré sa zvyknú zapisovať v tvare aditívnej korekcie ako:

$$\overline{f}_m = \overline{f}_{m-1} + \lambda_m s_m,$$

kde s_m je nenulový N-rozmerný vektor definujúci smer posunu a λ_m je nenulový skalár reprezentujúci dĺžku kroku.

V algoritme Gradient boostingu sa smer ($s_m = (s_{1m}, s_{2m}, \ldots, s_{Nm})^T$) volí ako smer najväčšieho spádu, ktorý predstavuje záporný gradient:

$$\begin{aligned} s_m &= -\nabla L(\overline{f}_m) \\ s_{im} &= -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x_i)=f_{m-1}(x_i)} \end{aligned}$$

Dĺžka kroku (λ_k) sa následne určuje optimalizáciou, konkrétne hľadaním jednorozmerného voľného extrému v smere záporného gradientu:

$$\begin{aligned} \underset{\lambda}{min} \quad & L(\overline{f}_m + \lambda s_m) \\ & \lambda \in \mathbb{R} \end{aligned}$$

Finálne riešenie tohto optimalizačného procesu možno po M iteráciách zapísať v tvare:

$$\overline{f}_M = \overline{f_o} + \sum_{m=1}^{M} -\lambda_m s_m,$$

kde $\overline{f}_0$ je počiatočný odhad modelu.

Všimnime si štrukturálnu zhodu medzi iteračným krokom odvodeným z metód voľnej optimalizácie ($\overline{f}_m = \overline{f}_{m-1} + \lambda_m \mathbf{s}_m$) a aktualizáciou nášho aditívneho modelu

$(f_m(x) = f_{m-1}(x) + T(x, \omega_m))$. V ideálnom prípade by sme chceli, aby nový bázický strom priamo reprezentoval optimálny posun, teda aby jeho predikcie spĺňali rovnosť $T(x_i, \omega_m) = \lambda_m s_{im}$ pre všetky pozorovania.

Zásadný rozdiel však spočíva v matematickom priestore, v ktorom operujeme. Kým vektor záporného gradientu $\mathbf{s}_m$ predstavuje nezávislé a neobmedzené hodnoty vyčíslené výlučne pre N trénovacích vzoriek, rozhodovací strom $T(x, \omega_m)$ je funkcia definovaná na celom priestore premenných. Navyše, predikcie stromu:

$$t_m = (T(x_1, \omega_m), T(x_2, \omega_m), \ldots, T(x_N, \omega_m))^T$$

nie sú nezávislé. Strom musí priradiť rovnakú hodnotu všetkým pozorovaniam, ktoré spadnú do spoločného listu. Taktiež gradient je pre nové, nevidené dáta nevyčísliteľný. Hlavnou úlohou ľubovolného predikčného modelu je schopnosť predikcie na nových dátach, čo by v prípade využitia numerického modelu nebolo možné.

Tento problém Gradient Boosting rieši tak, že v m-tom kroku nafituje rozhodovací strom $T(x, \omega_m)$, ktorého predikcie budú čo najviac aproximovať vektor záporného gradientu $\mathbf{s}_m$. Keďže ide o aproximáciu spojitých hodnôt gradientu, algoritmus využíva regresný strom, pričom na hľadanie optimálnych disjunktných regiónov minimalizuje štvorcovú odchýlku (MSE) od tohto gradientu:

$$H(Q_v) = \frac{1}{n_v} \sum_{x_i \in Q_v} (s_{im} - \overline{s}_m)^2$$

Vyššie opísaný proces Gradient Boostingu možno rozšíriť na riešenie problémov klasifikácie do K tried. Tento problém sa rieši trénovaním K samostatných binárnych klasifikátorov. Aby sme previedli našu K-klasifikáciu na binárnu musíme transformovať cieľovú premennú $y = (y_1, y_2, \ldots, y_N)^T$, kde $y_i \in \{1, 2, \ldots, K\}$. Zadefinujeme upravenú premennú ako $\tilde{y}$, kde:

$$\tilde{y} = (\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_K) = \begin{pmatrix} \tilde{y}_{11} & \tilde{y}_{12} & \cdots & \tilde{y}_{1K} \\ \tilde{y}_{21} & \tilde{y}_{22} & \cdots & \tilde{y}_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{y}_{N1} & \tilde{y}_{N2} & \cdots & \tilde{y}_{NK} \end{pmatrix}$$

$$\tilde{y}_{ik} \begin{cases} 1 & \text{ak } y_i = k \\ 0 & \text{ak } y_i \neq k \end{cases}$$

Pre každú z tried k sa následne buduje samostatný model $\tilde{f}_k$. Hoci ide o riešenie klasifikačného problému, jednotlivé bázické stromy sú z podstaty algoritmu regresné, keďže ich cieľom je aproximácia spojitých hodnôt gradientov. Stratová funkcia pre túto úlohu je definovaná ako:

$$L(\{\tilde{y}_{ik}, \tilde{f}_k(x_i)\}_{k=1}^{K}) = -\sum_{k=1}^{K} \tilde{y}_{ik} log(p_k(x_i)),$$

kde pravdepodobnosť $p_k(x_i)$ je získaná aplikáciou funkcie softmax na surové výstupy modelov:

$$p_k(x_i) = \frac{e^{\tilde{f}_k(x_i)}}{\sum_{l=1}^{K} e^{\tilde{f}_l(x_i)}}$$

Algoritmus učenia Gradient Boostingu pre K-triedovú klasifikáciu pozostáva z nasledujúcich krokov:

1. **Inicializácia:** Zvolíme počiatočné modely pre každú triedu vyriešením optimalizačnej úlohy:

$$\tilde{f}_{0k} = \underset{\gamma_k}{argmin} \sum_{i=1}^{N} L(\tilde{y}_{ik}, \gamma_k)$$

.

2. Pre m = 1 do M:

   (a) Vypočítame aktuálne pravdepodobnosti $p_{m-1,k}(x_i) = \frac{e^{\tilde{f}_{mk}(x_i)}}{\sum_{l=1}^{K} e^{\tilde{f}_{ml}(x_i)}}$ pre $i = 1, 2, \ldots, N$ a $k = 1, 2, \ldots, K$.

   (b) Vypočítame gradienty pre všetky pozorovania a triedy:
   $s_{ik} = -\left[\frac{\partial L(\{\tilde{y}_{il}, \tilde{f}_l(x_i)\}_{l=1}^{K})}{\partial \tilde{f}_k(x_i)}\right]_{\tilde{f}_l(x_i)=f_{m-1,k}(x_i)} = \tilde{y}_{ik} - p_{m-1,k}(x_i)$

   (c) Na vypočítaných gradientoch natrénujeme K regresných rozhodovacích stromov, čím získame rozdelenie do disjunktných regiónov R_{jmk}, (pre $j = 1, 2, \ldots, J_m$ a $k = 1, 2, \ldots, K$)

   (d) Pre každý región vypočítame optimálnu konštantu (váhu listu). Nech $S_{jmk} = \{i \mid x_i \in R_{jmk}\}$:

$$\gamma_{jmk} = \underset{\gamma}{argmin} \sum_{i \in S_{jmk}} L(\tilde{y}_{ik}, \tilde{f}_{m-1,k}(x_i) + \gamma)$$

(e) Aktualizujeme model pre každú triedu:
$$\tilde{f}_{m,k} = \tilde{f}_{m-1,k} + \sum_{j=1}^{J_{mk}} \gamma_{jmk} I(x \in R_{jmk})$$

3. **Výstup:** Finálny súbor modelov:
$\tilde{f}_k(x) = \tilde{f}_{M,k}(x)$

3.2.4 XGBoost

Pri boostingu sme predstavili štruktúru aditívneho rozvoja pre rozhodovacie stromy a definovali sme optimalizačnú úlohu pre hľadanie ich parametrov. V algoritme XGBoost (Extreme Gradient Boosting) sa táto formulácia rozširuje zavedením explicitného regularizačného člena, ktorý penalizuje komplexnosť modelu. Nová optimalizačná úloha má nasledovný tvar:

$$\underset{\{\omega_m\}_1^M}{min} \sum_{i=1}^{N} L(y_i, \sum_{m=1}^{M} T(x_i, \omega_m)) + \sum_{m=1}^{M} \rho(f_m),$$

kde penalizačný člen je definovaný $\rho(f) = \psi J + \frac{1}{2}\xi\|c\|^2$, pričom J je počet listov (disjunktných oblastí) stromu a vektor $c = (c_1, c_2, \ldots, c_J)$ predstavuje predikcie v jednotlivých listoch.

Predpokladáme, že stratová funkcia je diferencovateľná a konvexná. Penalizačný člen ρ má za úlohu kontrolovať zložitosť jednotlivých bazických stromov, čím model predchádza pretrénovaniu. Rovnako ako pri prístupe dopredného po-krokového aditívneho modelovania (Forward Stagewise Additive Modellingu), aj tu aproximujeme riešenie optimalizačnej úlohy sekvenčným pridávaním nových rozhodovacích stromov. V m-tom kroku hľadáme parametre nového stromu riešením úlohy:

$$\omega_m = \underset{\omega}{argmin} \left[\sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + T(x_i, \omega)) \right] + \rho(T(x, \omega))$$

Aplikáciou Taylorovho rozvoja druhého rádu na stratovú funkciu získame aproximáciu účelovej funkcie v tvare:

$$\sum_{i=1}^{N} \left[L(y_i, f_{m-1}(x_i)) + g_i T(x_i, \omega) + h_i T^2(x_i, \omega) + \mathcal{O}(T^3(x_i, \omega)) \right] + \rho(T(x, \omega))$$

kde prvá a druhá derivácia stratovej funkcie sú definované ako:

$$g_i = \frac{\partial L(y_i, f_{m-1}(x_i))}{\partial f_{m-1}(x_i)} \quad \text{a} \quad h_i = \frac{\partial^2 L(y_i, f_{m-1}(x_i))}{\partial f_{m-1}^2(x_i)}$$

Z pohľadu optimalizácie nového rozhodovacieho stromu v kroku m je prvý člen účelovej funkcie ($L(y_i, f_{m-1}(x_i))$) konštantný, keďže závisí iba od predikcií z predchádzajúcich krokov. Po jeho zanedbaní a odstránení členov vyšších rádov nadobudne optimalizačná úloha tvar:

$$\omega_m = \underset{\omega}{argmin} \sum_{i=1}^{N} \left[g_i T(x_i, \omega) + h_i T^2(x_i, \omega) \right] + \rho(T(x, \omega))$$

Rozpísaním penalizačného člena a rozhodovacích stromov v účelovej funkcii dostaneme:

$$\begin{aligned} \text{učelová f.} \quad &= \sum_{i=1}^{N} g_i \sum_{j=1}^{J} c_j I(x_i \in R_j) + h_i \sum_{j=1}^{J} c_j^2 I(x_i \in R_j) + \psi J + \tfrac{1}{2}\xi \sum_{j=1}^{J} c_j^2 \\ &= \sum_{j=1}^{J} \left[\left(\sum_{i \in S_j} g_i \right) c_j + \tfrac{1}{2} \left(\sum_{i \in S_j} h_i + \xi \right) c_j^2 \right] + \psi J, \end{aligned}$$

kde $S_j = \{i | x_i \in R_j\}$ označuje množinu indexov pozorovaní patriacich do j-tého listu. V druhom riadku rovnice sme zmenili poradie sčítavania z pozorovaní na listy, čo je možné vďaka tomu, že všetky pozorovania v rovnakom liste zdieľajú identickú predikciu c_j. Zadefinovaním sumy gradientov $G_j = \sum_{i \in S_j} g_i$ a sumy hessiánov $H_j = \sum_{i \in S_j} h_i$ pre každý list, možno účelovú funkciu zapísať v kompaktnom tvare premennej c_j:

$$\sum_{j=1}^{J} \left[G_j c_j + \frac{1}{2}(H_j + \xi) c_j^2 \right] + \psi J$$

Podobne ako pri izolovaných rozhodovacích stromoch, aj tu je cieľom určiť optimálne parametre listov c_j a štruktúru disjunktných regiónov R_j. Stanovenie optimálnych váh c_j pri už zafixovanej štruktúre regiónov R_j je analyticky triviálne. Keďže účelová funkcia predstavuje pre každý list samostatnú konvexnú kvadratickú funkciu (parabolu), jej minimum získame riešením nasledujúcej úlohy:

$$\min_{c} \sum_{j=1}^{J} \left[G_j c_j + \frac{1}{2}(H_j + \xi) c_j^2 \right] + \psi J$$

Položením prvej derivácie podľa c_j rovnej nule dostávame optimálne riešenie:

$$c_j^* = -\frac{G_j}{H_j + \xi}$$

Tento výsledok odhaľuje teoretickú pointu algoritmu XGBoost: hodnota c_j^* má presnú štruktúru ako pri Newtonovej metóde definovanej v [9]. Zatiaľ čo klasický

Gradient Boosting určuje krok iba na základe prvej derivácie (metóda najväčšieho spádu), XGBoost využíva informáciu aj o druhej derivácii (Hessián H_j).

Hodnota účelovej funkcie v bode optima ($-\frac{1}{2}\sum_{j=1}^{J}\frac{G_j^2}{H_j+\xi}+\psi J$) predstavuje metriku, ktorá kvantifikuje, aké kvalitné je aktuálne rozdelenie do regiónov R_j. Čím je táto hodnota nižšia, tým strom lepšie aproximuje optimálny krok na trénovacích dátach. V algoritme XGBoost je táto metrika podobná funkcii nečistoty uzla definovanej v časti 3.2.1.

Na trénovanie (konštrukciu) stromu v m-tom kroku aditívneho rozvoja sa využíva analogická logika ako pri štandardných rozhodovacích stromoch (algoritmus CART popísaný v časti 3.2.1).

Proces začína inicializáciou v aktuálnom uzle v. Pre všetky prípustné premenné sa diskretizuje priestor definovaním všetkých kandidátskych bodov rezu. Pre každé potenciálne rozdelenie množiny pozorovaní Q_v na ľavý pod-uzol (Q_v^{left}) a pravý pod-uzol (Q_v^{right}) sa kvantifikuje zníženie účelovej funkcie, tzv. Zisk (Gain), podľa vzorca:

$$\frac{1}{2}\left[\frac{G_{Lv}^2}{H_{Lv}+\xi}+\frac{G_{Rv}^2}{H_{Rv}+\xi}-\frac{G_v^2}{H+\xi}\right]-\psi, \tag{4}$$

kde G_v a H_v predstavujú súčet gradientov a hessiánov všetkých pozorovaní v pôvodnom nerozdelenom uzle Q_v. Hodnoty G_{Lv} a H_{Lv} zodpovedajú súčtom pre pozorovania, ktoré po rozdelení padnú do ľavej vetvy (Q_v^{left}), a analogicky G_{Rv} a H_{Rv} sú súčty pre pozorovania v pravej vetve (Q_v^{right}).

Optimálne rozdelenie uzla sa určí výberom rezu, ktorý maximalizuje hodnotu rovnice (4). Ak je maximálny dosiahnuteľný Zisk záporný (napríklad vplyvom silnej penalizácie ψ), uzol sa ďalej nedelí a stáva sa terminálnym listom. Zvyšok algoritmu, vrátane rekurzívneho delenia a aplikácie ďalších ukončovacích kritérií, pokračuje identicky, ako bolo definované v časti 3.2.1.

V prípade K-klasifikácie sa postupuje koncepčne rovnako ako pri štandardnom algoritme Gradient Boostingu. V každej iterácii m sa paralelne konštruuje K samostatných regresných stromov.

3.3 Kritériá kvality klasifikačných modelov

Táto podkapitola definuje matematický význam a interpretáciu hodnotiacich metrík, ktoré sú aplikované pri analýze výsledkov modelov logistickej regresie a XGBoostu.

Základným evaluačným výstupom klasifikačných modelov je klasifikačný report a matica zámien. Príklad takéhoto výstupu pre trénovaciu aj testovaciu množinu dát je znázornený na Obr. 28. Klasifikačný report kvantifikuje úspešnosť modelu pre každú z predikovaných tried (v kontexte tejto práce ide o štyri geografické regióny) prostredníctvom štyroch kľúčových ukazovateľov:

Precíznosť (Precision)

Predstavuje pomer správne klasifikovaných pozorovaní danej triedy ku všetkým pozorovaniam, ktorým model túto triedu priradil. Ak napríklad model označí 100 entít ako región Amerika, pričom v skutočnosti do tohto regiónu patrí len 60 z nich, precíznosť pre túto triedu je 0.60.

Úplnosť (Recall)

Predstavuje podiel správne klasifikovaných pozorovaní voči všetkým skutočným pozorovaniam patriacim do tejto triedy. Ak sa v dátach nachádza 100 entít z Ameriky a model z nich úspešne identifikuje 70 (pričom zvyšných 30 priradí iným regiónom), úplnosť dosahuje hodnotu 0.70.

F1-skóre (F1-Score)

Predstavuje harmonický priemer medzi presnosťou a úplnosťou. Formálne je definované vzťahom:

$$F_1 = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}$$

Táto metrika poskytuje komplexnejší pohľad na výkonnosť modelu, nakoľko penalizuje extrémne rozdiely medzi precíznosťou a úplnosťou.

Veľkosť tried (Support)

Udáva absolútny počet skutočných výskytov danej triedy v posudzovanom datasete. Táto hodnota nereprezentuje počet predikcií modelu, ale reálnu distribúciu dát.

V spodnej časti klasifikačného reportu sa nachádzajú globálne ukazovatele výkonnosti modelu:

- **Celková presnosť (Accuracy):** Vyjadruje celkový podiel správne klasifikovaných pozorovaní naprieč všetkými triedami voči celkovej veľkosti datasetu.
- **Makro priemer (Macro Avg):** Predstavuje nevážený aritmetický priemer metrík pre všetky triedy. Každá trieda má pri výpočte rovnakú váhu bez ohľadu na jej veľkosť.
- **Vážený priemer (Weighted Avg):** Reprezentuje aritmetický priemer metrík, v ktorom je váhou pre každú triedu jej veľkosť. Tento priemer zohľadňuje triednu nevyváženosť datasetu.

Komplementárnym analytickým nástrojom je matica zámien (confusion matrix, Obr. 28 dole). Táto matica obsahuje zložky C_{ij}, kde riadok i predstavuje skutočnú triedu a stĺpec j predstavuje predikovanú modelom. Pre účely tejto práce sme zvolili riadkovo normalizovanú (relatívnu) formu matice. Zložku C_{11} s hodnotou 0.76 napríklad interpretujeme tak, že z celkového počtu 313 pozorovaní skutočne patriacich do triedy Amerika model správne identifikoval 76 %. Z tohto matematického zápisu vyplýva, že hodnoty na diagonále matice sú identické s metrikou úplnosti (recall) pre jednotlivé triedy. Na rozdiel od klasifikačného reportu však matica zámien poskytuje detailnú vizualizáciu chýb, exaktne ukazuje, do ktorých nesprávnych kategórií model pozorovania najčastejšie priraďuje.

Na hlbšiu analýzu spoľahlivosti pravdepodobnostných predikcií modelu bol implementovaný vlastný hodnotiaci nástroj vo forme kvantilového kalibračného grafu (Obr. 29). Pred interpretáciou grafu je nevyhnutné zadefinovať pojem „istota modelu“. Istota je v tomto kontexte chápaná ako pravdepodobnosť, s akou model priradí pozorovaniu zvolenú triedu. Ak sa na to pozrieme z iného pohľadu, tak predstavuje maximálnu hodnotu pravdepodobnosti (získanej prostredníctvom funkcie Softmax), na základe ktorej algoritmus vykonáva finálnu predikciu.

Metodika konštrukcie kalibračného grafu spočíva v agregácii predikcií. Všetky pozorovania sú vzostupne zoradené podľa istoty modelu a následne rozdelené do piatich ekvidištančných kvantilov, pričom každý obsahuje presne 20% dát. Pre každú z týchto piatich skupín je vypočítaný bod, ktorého súradnica x predstavuje priemernú predikovanú istotu modelu v danej skupine, a súradnica y predstavuje priemernú

presnosť (podiel správnych klasifikácií) v tej istej skupine. Táto exaktná vizualizácia umožňuje detailne sledovať a hodnotiť, či miera istoty klasifikátora reálne zodpovedá jeho skutočnej chybovosti.

4 Aplikácia klasifikačných modelov na dáta

Táto kapitola je venovaná procesu trénovania, optimalizácie a následnej interpretácii výsledkov predikčných modelov logistickej regresie a XGBoost. Modely boli aplikované na dataset vybudovaný a predspracovaný v rámci Kapitoly 2. Pred samotným trénovaním bola na dátach vykonaná dodatočná filtrácia: z datasetu boli odstránené všetky entity (používatelia), ktoré vykazovali celkovú históriu menšiu ako 15 transakcií (zostatok dát po tomto kroku bol 1942). Predpokladom tohto kroku je, že pri takomto nízkom počte interakcií nie je možné štatisticky spoľahlivo identifikovať relevantný časový vzor transakčnej aktivity. Takéto pozorovania by v procese učenia pôsobili ako informačný šum, čo odôvodňuje ich exklúziu z finálneho datasetu. Sme si vedomí, že stanovenie tejto hranice nevylúči všetky pozorovania, ktoré môžu pôsobiť ako šum, ale dúfame, že klasifikačné modely takýmto pozorovaniam budú priraďovať výrazne nižšiu istotu predikcie.

Predikčná schopnosť jednotlivých modelov je vyhodnocovaná primárne prostredníctvom porovnania výkonnosti na trénovacej a testovacej množine dát. Pre rozdelenie datasetu bol zvolený pomer 75:25 (75% trénovacia množina, 25% testovacia množina). Delenie bolo realizované s využitím stratifikovaného výberu, čím sa zabezpečilo, že distribúcia a proporcionálne zastúpenie jednotlivých tried (regiónov) zostali v oboch množinách exaktne zachované.

S cieľom zabezpečiť striktnú reprodukovateľnosť prezentovaných výsledkov bola pre generátor náhodného výberu trénovacej a testovacej vzorky zafixovaná počiatočná hodnota[3]. Všetky nasledujúce analýzy, grafické výstupy a interpretácie výsledkov modelov sú priamo naviazané na toto konkrétne rozdelenie dát.

4.1 Analýza výsledkov použitia logistickej regresie

Analýzu výsledkov otvárame predstavením logistickej regresie, ktorú v rámci tejto práce považujeme za referenčný model. Hlavným konfiguračným parametrom boli váhy tried, pričom každému pozorovaniu patriacemu do triedy $k \in \{1, 2, 3, 4\}$ bola

[3]parameter random_state = 7736

priradená váha w_k podľa vzťahu:

$$w_k = \frac{N}{K \cdot n_k},$$

kde N predstavuje celkový počet pozorovaní v trénovacej množine, K je počet tried a n_k je počet pozorovaní v triede k. Okrem váh bol v rámci algoritmu definovaný maximálny počet iterácií potrebný na dosiahnutie konvergencie pri hľadaní optimálnych parametrov vektora β.

Na obrázku 28 je zobrazený klasifikačný report spolu s maticou zámien pre trénovaciu (vľavo) a testovaciu množinu dát (vpravo). Pri analýze klasifikačného reportu pre región Centrálna Ázia na trénovacej vzorke vidíme, že precíznosť dosahuje 51%, čo znamená, že len približne polovica pozorovaní priradených do tejto triedy bola klasifikovaná správne. Model však úspešne identifikoval (recall) 64% všetkých skutočných prípadov tohto regiónu. Opačný trend vykazuje región Východná Ázia, kde precíznosť dosahuje 69%, no úplnosť klesá na 56 %. Z týchto hodnôt vyplýva, že model sa pri klasifikácii Východnej Ázie správa konzervatívnejšie, zatiaľ čo v prípade Centrálnej Ázie má tendenciu túto triedu priraďovať častejšie.

Z matice zámien pre trénovaciu množinu vyplýva, že dáta zachytávajúce časovú aktivitu používateľov nesú signifikantnú informáciu o ich geografickej polohe. Hoci sa celková presnosť modelu môže zdať nižšia, matica odhaľuje, že k chybám dochádza primárne pri zámene medzi susediacimi regiónmi. Najvýraznejšia nepresnosť nastáva medzi Centrálnou a Východnou Áziou: až 26% subjektov z Východnej Ázie bolo klasifikovaných ako Centrálna Ázia a 21 % z Centrálnej Ázie bolo priradených k Východnej. Tento jav je pravdepodobne spôsobený malým časovým posunom medzi hlavnými populačnými centrami týchto regiónov (Centrálna Ázia s UTC +5:30 vs. Východná Ázia s UTC +7:00). Takto nízky posun časovej zóny nemusí byť pre model dostatočným diskriminačným príznakom.

Výsledky na testovacej sade sú takmer identické s trénovacou sadou, čo naznačuje absenciu výrazného pretrénovania. To navyše potvrdzuje, že model sa neučí iba náhodný šum, ale časová aktivita používateľov skutočne obsahuje relevantné informácie.

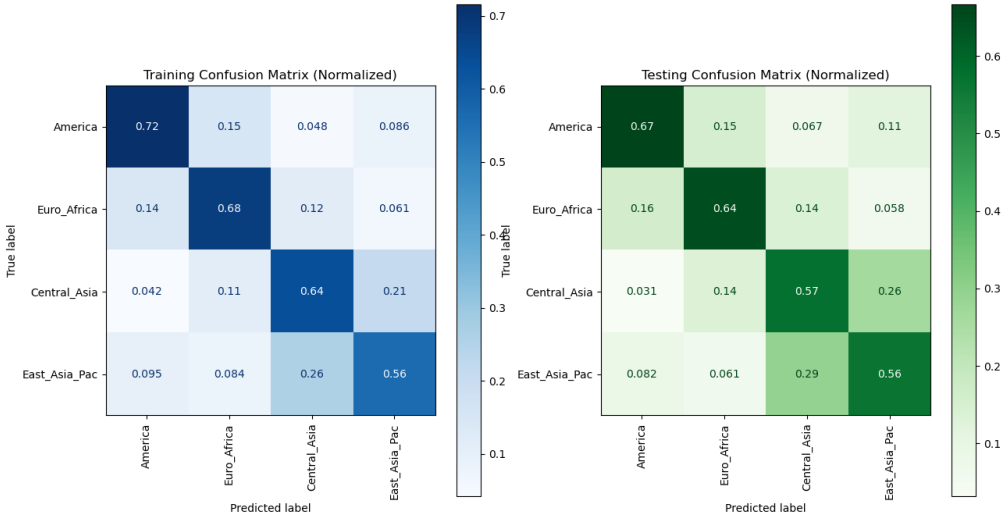
Cieľom práce je však získať model, ktorý vykazuje vysokú presnosť najmä pri pozorovaniach s vysokou predikovanou istotou. Z globálneho kalibračného grafu (Obr.

Classification Report (Train)

	precision	recall	f1-score	support
America	0.66	0.72	0.69	313
Central_Asia	0.51	0.64	0.56	288
East_Asia_Pac	0.69	0.56	0.62	442
Euro_Africa	0.71	0.68	0.69	413
accuracy			0.64	1456
macro avg	0.64	0.65	0.64	1456
weighted avg	0.65	0.64	0.64	1456

Classification Report (Test)

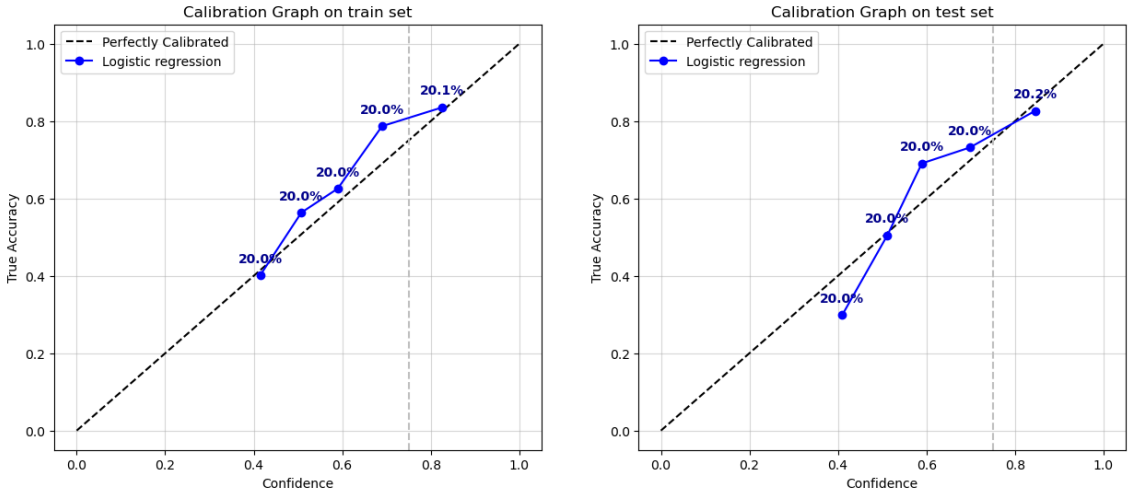
	precision	recall	f1-score	support
America	0.65	0.67	0.66	105
Central_Asia	0.44	0.57	0.50	96
East_Asia_Pac	0.65	0.56	0.60	147
Euro_Africa	0.70	0.64	0.67	138
accuracy			0.61	486
macro avg	0.61	0.61	0.61	486
weighted avg	0.62	0.61	0.61	486



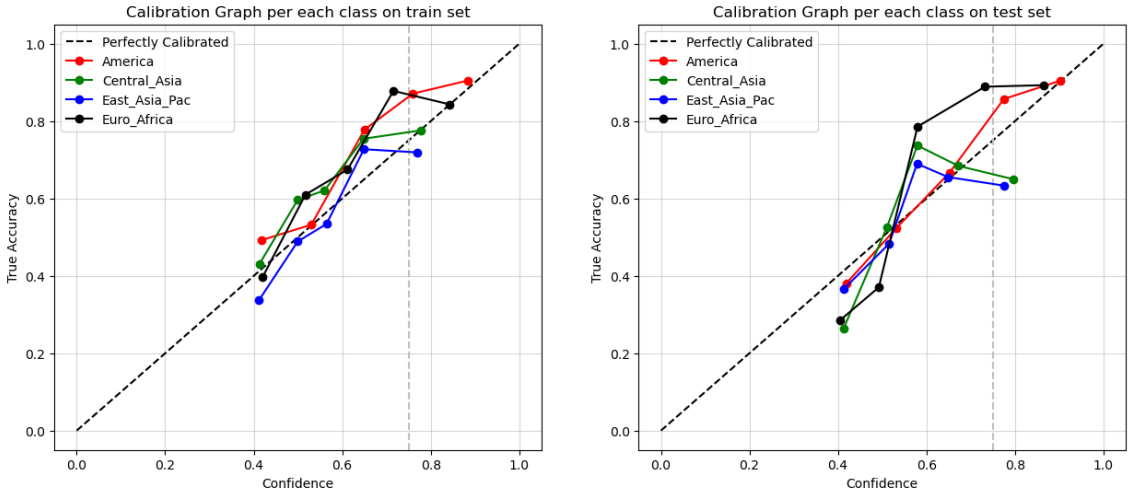
Obr. 28: Klasifikačný report a matica zámien logistickej regresie pre trénovaciu a testovaciu množinu

29) vyplýva, že model je na trénovacích dátach takmer ideálne nakalibrovaný. Na testovacej sade však pri najnižšom kvantile istoty (dolných 20% dát) dochádza k miernemu poklesu priemernej presnosti, čo však nepredstavuje výrazný nedostatok modelu.

Detailnejší pohľad na jednotlivé triedy (Obr. 30) odhaľuje, že tento pokles spôsobujú najmä regióny Centrálna a Východná Ázia. Pri 20% najistejších pozorovaní je priemerná presnosť relatívne nízka, približne 65% pre Východnú Áziu aj pre Centrálnu Áziu, zatiaľ čo priemerná istota pri tejto skupine sa pohybuje okolo 80%. Hoci tento najvyšší kvantil vykazuje pre ázijské regióny podstatne nižšiu priemernú istotu v porovnaní s regiónmi Euroafrika a Amerika, skutočná presnosť modelu je aj tak nedostatočná vzhľadom na mieru predikovanej istoty. Tieto nedostatky sa pokúsime eliminovať v ďalšej časti práce využitím robustnejšieho algoritmu XGBoost. Záverečná analýza chybovosti (Obr. 31 a 32) zobrazuje distribúciu predikcií pre reálne triedy Centrálna a Východná Ázia v závislosti od istoty modelu. Zaujímavým zistením je, že v prípade Centrálnej Ázie model pri narastajúcej neistote často zamieňa

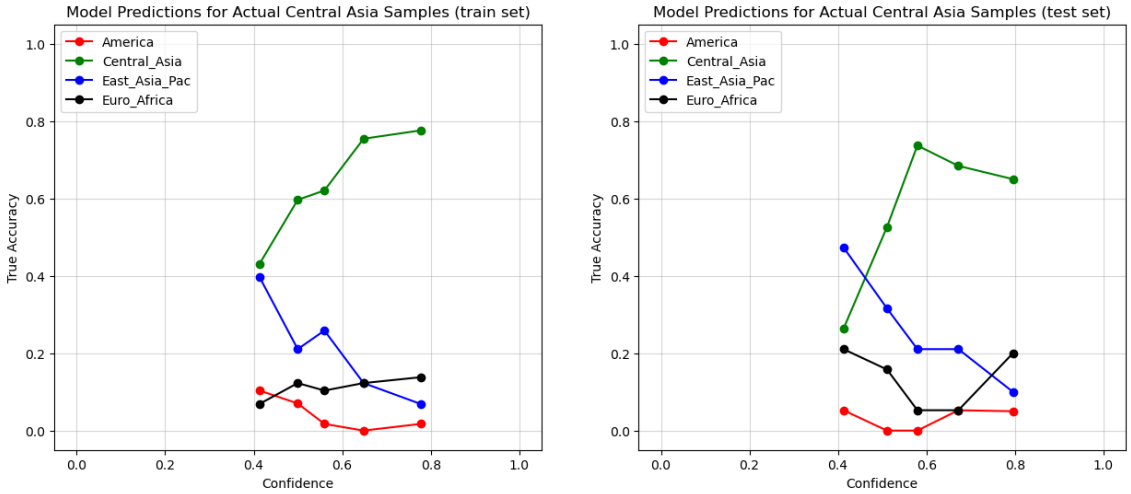


Obr. 29: Globálny kalibračný graf logistickej regresie

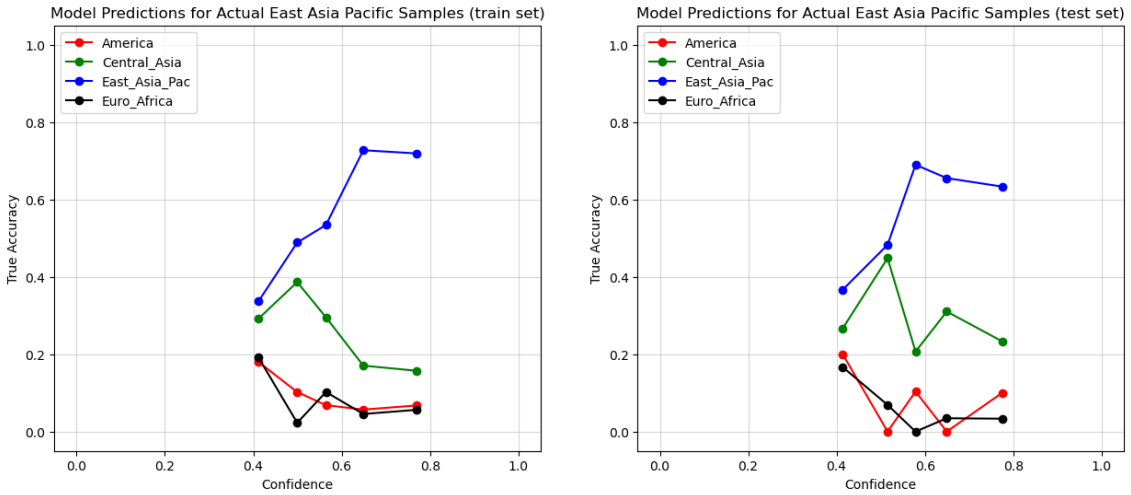


Obr. 30: Kalibračný graf logistickej regresie rozdelený podľa cieľových regiónov

tento región za Euroafriku, a nie za Východnú Áziu, ako by sa dalo predpokladať z matice zámien. Pri reálnej triede Východná Ázia je však najčastejšia zámena za Centrálnu Áziu dominantná naprieč celým spektrom istoty, čo korešponduje s predpokladom o geografickej a časovej blízkosti.



Obr. 31: Distribúcia predikcií pre reálnu triedu Centrálna Ázia



Obr. 32: Distribúcia predikcií pre reálnu triedu Východná Ázia

4.2 Hľadanie optimálnych parametrov XGBoostu

Ako vyplynulo z predchádzajúcej analýzy, logistická regresia vykazuje pri pozorovaniach s vysokou predikovanou istotou mierne nedostatky pri predikcii Centrálnej a

Východnej Ázii. Tento jav sa pokúsime eliminovať aplikáciou robustnejšieho algoritmu XGBoost a optimalizáciou jeho hyperparametrov. Hoci implementácia tohto algoritmu v knižnici xgboost poskytuje rozsiahle možnosti konfigurácie, v tejto práci sme sa zamerali na optimalizáciu nasledujúcej podmnožiny kľúčových parametrov:

- **n_estimators:** Vyjadruje počet rozhodovacích stromov v aditívnom rozvoji (M).
- **eta:** Predstavuje rýchlosť učenia (learning rate), teda váhu, ktorou jednotlivé stromy prispievajú do aditívneho rozvoja. Účelová funkcia definovaná v časti 3.2.4 bude vyzerať nasledovne: $\left[\sum_{i=1}^{N} L(y_i, f_{m-1}(x_i) + \boldsymbol{\eta} T(x_i, \omega))\right] + \rho(T(x, \omega))$
- **gamma:** Minimálna redukcia stratovej funkcie potrebná na vytvorenie ďalšieho uzla. Identické so ψ definovaným v časti 3.2.4
- **reg_lambda:** Regularizačný člen aplikovaný na váhy listov. Identické so ξ definovaným v časti 3.2.4
- **max_depth:** Maximálna povolená hĺbka rozhodovacieho stromu.
- **min_child_weight:** Podobne ako pri klasifikačných stromoch ide o minimálny počet vzoriek v liste, avšak v kontexte algoritmu XGBoost je tento parameter definovaný ako minimálna suma hesiánov (h_i) všetkých pozorovaní v liste.
- **subsample:** Pomer pre náhodný výber pozorovaní z intervalu $(0, 1]$. Hodnota 0.5 znamená, že algoritmus pri konštrukcii každého stromu náhodne vzorkuje 50% dát z trénovacej množiny, čo predchádza pretrénovaniu..
- **colsample_bytree:** Pomer náhodne vybraných príznakov (premenných) pri konštrukcii každého stromu.

Optimalizácia hyperparametrov bola realizovaná prostredníctvom 5-násobnej krosvalidácie (5-fold cross-validation) so zachovaním pomeru rozdelenia tried. Pri tomto procese sa trénovacia množina dát rozdelí na 5 disjunktných podmnožín, pričom model sa iteratívne trénuje na štyroch z nich a jeho výkonnosť sa vyhodnotí na piatej (validačnej) množine. Na efektívne prehľadávanie priestoru parametrov bol namiesto

výpočtovo neefektívneho diskretizačného prístupu,pri ktorom by algoritmus testoval všetky kombinácie vopred zvolených hodnôt, využitý framework Optuna. Ten využíva princípy optimalizácie, vďaka čomu algoritmus inteligentne odhaduje smer, v ktorom úprava parametrov s najvyššou pravdepodobnosťou zlepší sledované kvalitatívne kritérium. Podrobný matematický aparát tohto princípu je popísaný v štúdii [1], pričom v tejto práci sa kvôli rozsahu nebudeme venovať jeho detailom.

Priestor vyhľadávania bol pre optimalizačný algoritmus exaktne ohraničený nasledujúcimi intervalmi: $\mathbf{n_estimators} \in [100, 1000] \cap \mathbb{Z}$, $\mathbf{eta} \in [0.01, 0.1]$, $\mathbf{gamma} \in [0, 5]$, $\mathbf{reg_lambda} \in [0, 5]$, $\mathbf{max_depth} \in [2, 8] \cap \mathbb{Z}$, $\mathbf{min_child_weight} \in [1, 20] \cap \mathbb{Z}$, $\mathbf{subsample} \in [0.6, 1]$ a $\mathbf{colsample_bytree} \in [0.6, 1]$.

Ako hlavné kritérium kvality modelu bolo zvolené Brierovo skóre, formálne definované ako:

$$\text{brier_score} = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} (I(y_i = k) - p_{ik})^2,$$

kde p_{ik} je modelom priradená pravdepodobnosť zaradenia i-teho pozorovania do triedy k. Táto metrika bola preferovaná z dôvodu, že výrazne penalizuje model za nesprávne predikcie spojené s vysokou mierou istoty. Napríklad v situácii, kedy model nesprávne klasifikuje pozorovania a danej triede priradí pravdepodobnosť 91% a zvyšným trom triedam 3%, Brierovo skóre bude neúmerne vysoké (1.7708). Naopak, pri správnej klasifikácii s rovnakou istotou je penalizácia len minimálna (0.0108). Ak by model rovnomernejšie rozložil neistotu (napríklad nesprávnej triede by priradil 40% a ostatným po 20%), Brierovo skóre by dosiahlo hodnotu 0.8800. Ako je možné vidieť, miera penalizácie je pri priznanej neistote podstatne miernejšia v porovnaní s prílišnou a chybnou istotou modelu.

Základná minimalizácia priemerného Brierovho skóre (resp. maximalizácia jeho zápornej hodnoty) však počas krosvalidácie viedla k vzniku pretrénovaných výsledkov na validačnej množine. Pre zabezpečenie lepšej generalizácie bolo preto Brierovo skóre doplnené o dodatočný penalizačný člen, ktorý sankcionuje rozdielnosť chýb medzi trénovacou a validačnou množinou. Finálne hodnotiace kritérium, ktoré sme prostredníctvom frameworku Optuna maximalizovali, je definované nasledovne:

$$-\text{brier_score}_{val} - \alpha(-\text{brier_score}_{train} + \text{brier_score}_{val}),$$

kde označenia reprezentujú priemerné Brierovo skóre vypočítané naprieč všetkými piatimi iteráciami krosvalidácie.

Pri fixácii penalizačného koeficientu na hodnotu $\alpha = 1$, kedy model vykazoval minimálne známky pretrénovania, konvergoval optimalizačný algoritmus k nasledujúcim parametrom: n_estimators = 211, eta = 0.0387, gamma = 4.9988, reg_lambda = 3.3169, max_depth = 2, min_child_weight = 19, subsample = 0.6351 a colsample_bytree = 0.8467. Samotné hodnoty (najmä vysoká úroveň regularizácie gamma, reg_lambda a prísne obmedzenia pre delenie max_depth a min_child_weight) exaktne dokazujú, že model je silne reštriktívny a striktne zabraňuje pretrénovaniu, čoho priamym dôkazom je aj skutočnosť, že väčšina generovaných stromov v rozvoji pozostáva len z troch alebo štyroch listov.

4.3 Analýza výsledkov použitia XGBoostu

V tejto časti práce sa zameriame na vyhodnotenie a porovnanie výsledkov optimalizovaného modelu XGBoost voči referenčnému modelu logistickej regresie. Analýzu otvárame vizualizáciou klasifikačného reportu a matice zámien (Obr. 33). Získané výsledky sú vo veľkej miere porovnateľné s výstupmi logistickej regresie, s výnimkou mierneho zlepšenia F1-skóre na trénovacej aj testovacej sade. Je zrejmé, že model naďalej naráža na identické problémy pri klasifikácii regiónov Centrálna a Východná Ázia.

Primárnym cieľom aplikácie algoritmu XGBoost však bola snaha o zvýšenie presnosti modelu pri pozorovaniach s vysokou predikovanou istotou. Z globálneho kalibračného grafu (Obr. 34) vyplýva, že oproti logistickej regresii došlo na testovacej sade k miernemu zlepšeniu predikcie v najvyššom kvantile (20% najistejších predikcií). Tento trend ešte zreteľnejšie potvrdzuje dekompozícia kalibračného grafu podľa jednotlivých regiónov (Obr. 35). Z grafu je evidentné plošné zvýšenie priemernej úspešnosti na testovacej sade pre horný kvantil istoty, a to špecificky v prípadoch Centrálnej Ázie, Východnej Ázie.

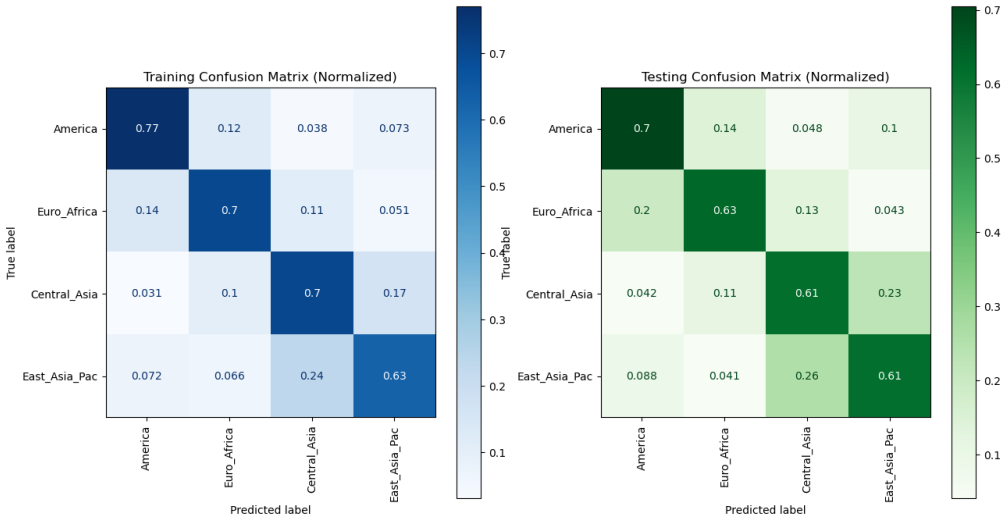
Detailnejší pohľad na distribúciu predikcií v závislosti od istoty modelu XGBoost (Obr. 36) ukazuje reálnu vzorku Východnej Ázie. Ako je možné vidieť, najčastejšou chybou pri Východnej Ázii zostáva jej zámena za Centrálnu Áziu. Ako už

Classification Report (Train)

	precision	recall	f1-score	support
America	0.71	0.77	0.74	313
Central_Asia	0.55	0.70	0.62	288
East_Asia_Pac	0.75	0.63	0.68	442
Euro_Africa	0.75	0.70	0.73	413
accuracy			0.69	1456
macro avg	0.69	0.70	0.69	1456
weighted avg	0.70	0.69	0.69	1456

Classification Report (Test)

	precision	recall	f1-score	support
America	0.63	0.70	0.66	105
Central_Asia	0.49	0.61	0.55	96
East_Asia_Pac	0.70	0.61	0.65	147
Euro_Africa	0.73	0.63	0.68	138
accuracy			0.64	486
macro avg	0.64	0.64	0.63	486
weighted avg	0.65	0.64	0.64	486

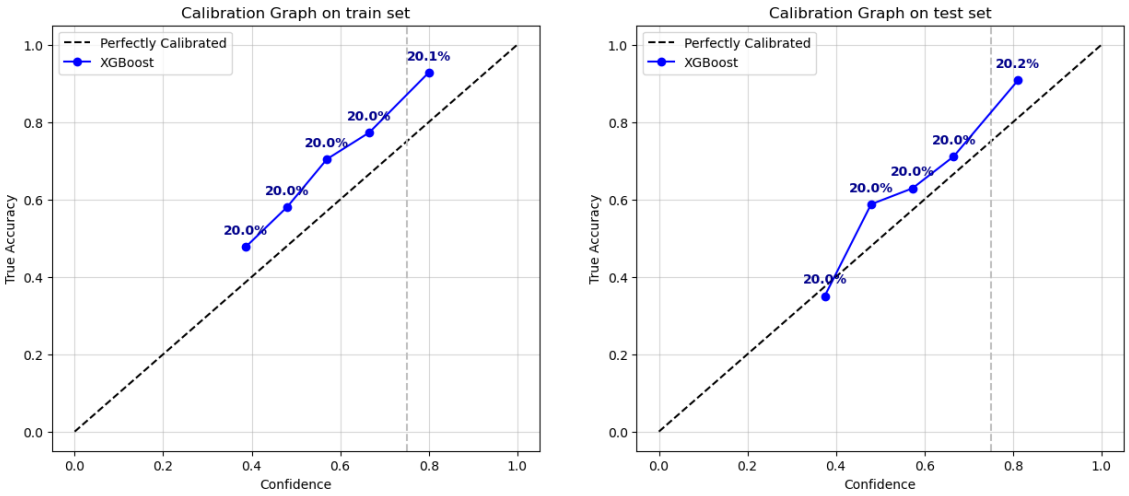


Obr. 33: Klasifikačný report a matica zámien modelu XGBoost pre trénovaciu a testovaciu množinu

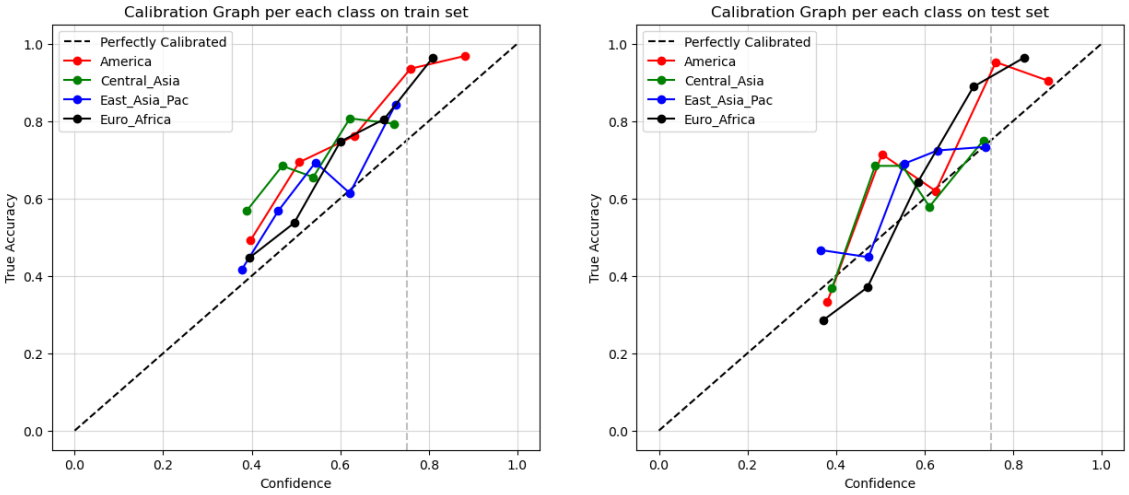
bolo konštatované pri analýze logistickej regresie, tento jav je s najväčšou pravdepodobnosťou spôsobený fázovou blízkosťou časových pásiem, ktoré tvoria dominantnú časť týchto dvoch regiónov (najmä Centrálna Ázia s UTC +5:30 a Východná Ázia s UTC +7:00). Vzhľadom na tento malý časový posun je zrejmé, že výlučne na základe vzorov časovej aktivity nedokáže ani komplexnejší model tieto dva regióny od seba spoľahlivo separovať.

Z celkového hľadiska priniesla aplikácia modelu XGBoost mierne zlepšenie výkonnosti oproti logistickej regresii. Avšak systémové problémy s prekrývajúcimi sa časovými pásmami, ktoré boli prítomné už v základnom modeli, sa nepodarilo úplne odstrániť, len čiastočne zlepšiť.

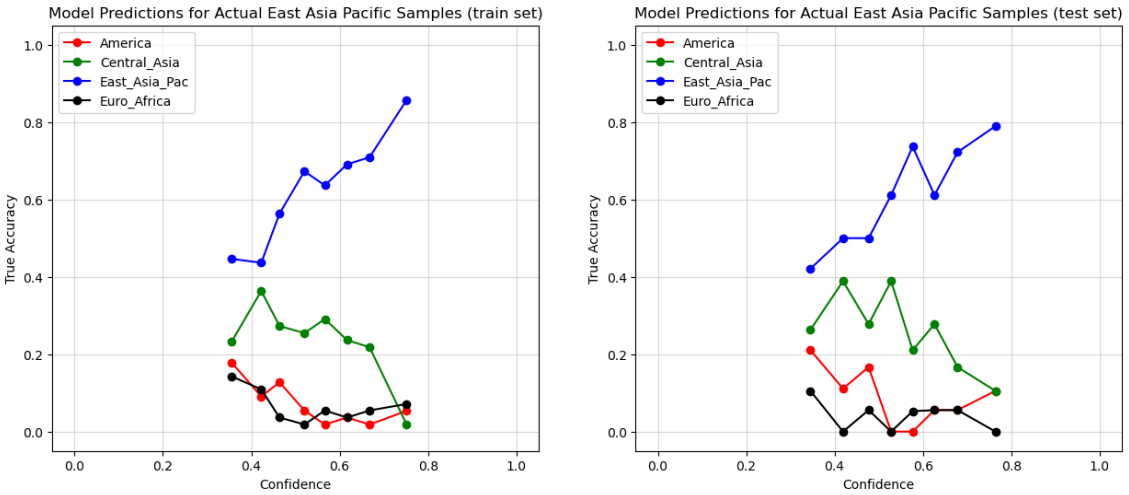
V záverečnej fáze analýzy sme pristúpili k manuálnemu vyhodnoteniu predikcií na testovacej vzorke (celkovo 486 pozorovaní). Vychádzame z predpokladu, že model pri klasifikácii deteguje primárne okno neaktivity (obdobie spánku), keďže ide o najvýraznejší rozlišovací znak naprieč regiónmi. V reálnych dátach je však vysoká pravdepodobnosť existencie jednotlivcov, ktorých biorytmus nekorešponduje



Obr. 34: Globálny kalibračný graf modelu XGBoost



Obr. 35: Kalibračný graf modelu XGBoost rozdelený podľa cieľových regiónov



Obr. 36: Distribúcia predikcií modelu XGBoost pre reálnu triedu Východná Ázia

so štandardným denným cyklom daného regiónu (napríklad používatelia spiaci cez deň a vykonávajúci transakcie v sieti Bitcoin v noci).

Pri manuálnej kontrole sme analyzovali predikcie, ktoré model klasifikoval s istotou 70 a viac percent. Týchto pozorovaní bolo 123 (25.3 % z celkovej testovacej sady), pričom model z nich nesprávne klasifikoval 14 (11.4 %). Detailné skúmanie týchto 14 chýb odhalilo, že iba v 2 až 3 prípadoch model zlyhal spôsobom, kedy mohol radšej zvoliť predikciu s nižšou istotou. Zvyšné nesprávne klasifikácie boli s najväčšou pravdepodobnosťou spôsobené anomálnym denným cyklom používateľov.

Pri znížení hranice istoty na 60 a viac percent model klasifikoval 214 pozorovaní (44 %), pričom k chybe došlo v 42 prípadoch (19.6 %). V tejto skupine bola zreteľná najmä častá zámena medzi Centrálnou a Východnou Áziou, čo potvrdzuje aj matica zámien. Pri kontrole týchto pozorovaní sme dospeli k záveru, že model tu naráža na svoje hranice, nakoľko z dôvodu silnej podobnosti časových vzorov medzi týmito regiónmi je ich exaktné rozlíšenie takmer nemožné.

Nakoniec sme pri pozorovaniach s istotou modelu nižšou ako 50 % skúmali, či model do tejto kategórie nezaradil používateľov s jasne definovaným a rozlíšiteľným oknom neaktivity. Počet takýchto prípadov bol však štatisticky zanedbateľný (identifikovali sme menej ako 10 pozorovaní). Tento fakt potvrdzuje, že nízka istota modelu reálne odráža absenciu jasného vzoru v dátach.

5 Semi-supervised learning

Semi-supervised learning predstavuje metódu strojového učenia, ktorá pri konštrukcii klasifikačných modelov efektívne kombinuje využitie obmedzeného množstva označených dát s veľkým objemom neoznačených dát. Ako už bolo v predchádzajúcich častiach tejto práce zdôraznené, získanie exaktnej geografickej lokality k entitám v sieti Bitcoin je metodologicky náročná úloha, ktorej bola venovaná značná časť tejto práce.

Na druhej strane, ak nie je primárnym cieľom identifikácia konkrétnej polohy, je možné získať dáta o všetkých transakciách realizovaných v sieti priamo z blockchainu. Kľúčovým krokom v tomto procese je filtrácia zmenární a iných rozsiahlych zhlukov, ktoré nereprezentujú individuálne správanie bežných používateľov. Jednoznačne však možno konštatovať, že pomer neoznačených dát k označeným je v praxi extrémne vysoký. Z tohto dôvodu je pre presnosť klasifikačných modelov prínosné, ak dokážu extrahovať relevantné informácie aj z neoznačených dát. Pri teoretickom opise metód semi-supervised learningu sme v tejto kapitole vychádzali primárne z vedeckých prác [17], [18], [19] a [5]. Pre účely experimentálnej časti práce budú predstavené najmä koncepty samoučenia (self-training) a šírenia označení (label propagation).

V rámci procesu zberu dát opísaného v Kapitole 2 vznikla sekundárna databáza adries a entít s platnou transakčnou históriou, pre ktoré však nebolo možné jednoznačne určiť geografickú lokalitu. Tieto subjekty tvoria základ neoznačeného datasetu. Proces spracovania týchto dát zahŕňal niekoľko krokov, identických ako boli použité v kapitole 2: zber údajov prostredníctvom blockchainových prehliadačov, exklúziu inštitucionálnych entít pomocou Arkham API, aplikáciu heuristického zhlukovania a následnú agregáciu histórie na úroveň entít. Po finálnej filtrácii entít s históriou nižšou ako 15 transakcií a odstránení duplicít pozostáva výsledná neoznačená vzorka z 9 837 subjektov.

Pre formálny opis algoritmov v tejto kapitole zavádzame nasledujúce označenie:

- $X_L \in \mathbb{R}^{N \times P}$ reprezentuje maticu vstupných označených (trénovacích) dát, ku ktorým prislúcha vektor cieľových tried $Y_L \in \{1, 2, \ldots, K\}^N$.

- $X_U \in \mathbb{R}^{S \times P}$ reprezentuje maticu neoznačených dát, kde pre cieľové hodnoty platí $Y_U = (-1, -1, \ldots, -1)^T$. Hodnota -1 v tomto kontexte indikuje absenciu informácie o príslušnosti k triede.

5.1 Samoučenie (Self-training)

Ako prvá metóda bola zvolená metóda samoučenia, ktorá predstavuje jeden z najjednoduchších princípov integrácie neoznačených dát do predikčných modelov. Algoritmus spočíva v úvodnom natrénovaní klasifikátora výlučne na množine označených dát (X_L). Tento referenčný model je následne aplikovaný na neoznačenú množinu dát (X_U) za účelom generovania predikcií. Pozorovania, pri ktorých model dosiahne najvyššiu mieru predikovanej istoty, sú spolu s ich predikovanými triedami pridané do pôvodnej trénovacej množiny. Model je následne trénovaný na tejto rozšírenej sade dát.

Ako uvádzajú autori v [17] a [18], hlavným rizikom tohto prístupu je propagácia a amplifikácia počiatočných chýb modelu. Ak model nesprávne klasifikuje neoznačené pozorovanie s vysokou istotou, táto chyba sa stáva trvalou súčasťou tréningového procesu a v ďalších iteráciách sa prehlbuje. Tento problém bol v práci proaktívne analyzovaný už pri optimalizácii referenčného modelu XGBoost (Kapitola 4), kde bola vyhodnotená jeho kalibrácia. Keďže predikcie pri najvyšších úrovniach istoty vykazovali uspokojivú spoľahlivosť na trénovacej aj testovacej množine, metóda samoučenia bola vyhodnotená ako potenciálne prínosná pre analyzovaný dataset.

Pre aplikáciu algoritmu bolo nevyhnutné definovať prahovú hodnotu istoty pre akceptáciu nových označení a celkový počet iterácií pridávania nových dát do označenej sady. Prostredníctvom 5-násobnej krosvalidácie bol testovaný efekt zmeny výkonnosti modelu oproti referenčnému XGBoostu pri rôznych kombináciách týchto parametrov. Ako primárne evaluačné metriky boli zvolené priemerné F1-skóre a Brierovo skóre merané na trénovacej a validačnej množine.

V Tabuľke 4 je kvantifikovaný vplyv zmeny prahovej hodnoty istoty (pri fixnom počte 10 iterácií) na celkovú výkonnosť modelu v porovnaní so základným, plne supervíznym modelom XGBoost. Z výsledkov vyplýva, že pri stanovení nízkej prahovej hodnoty istoty (75%) dochádza k signifikantnej degradácii modelu vo všetkých sle-

dovaných metrikách. Ani detailnejšia analýza klasifikačného reportu, matice zámien a kalibračných kriviek nepreukázala žiadnu pridanú informačnú hodnotu. Následné inkrementálne zvyšovanie prahovej hodnoty síce viedlo k postupnému zlepšovaniu výkonnosti (spolu s prirodzeným poklesom objemu presunutých dát do označených), avšak model v žiadnom testovanom scenári neprekonal svoj pôvodný referenčný stav.

Empirické zistenia naznačujú, že integrácia samoučenia v tomto konkrétnom prípade znižuje generalizačnú schopnosť algoritmu. Referenčný model XGBoost bol zjavne dostatočne presný s pôvodnými dátami. Pri manuálnej kontrole predikcii pôvodného modelu na neoznačenej sade dát sa nám predikcie javili ako presné. Našou ideou bolo presnejšie vyjadriť rozdiely medzi Centrálnou a Východnou Áziou. Ako však vidíme na výsledkoch model sa pridaním takýchto zhoršuje. Vo všeobecnosti je vplyv neoznačených dát v semi-supervised learningu náročné odhadnúť predom, pričom aplikácia základného samoučenia sa pre túto doménu ukázala ako kontraproduktívna. Hoci odborná literatúra (napr. [17]) predstavuje aj pokročilejšie varianty samoučenia, ich implementácia by z dôvodu absencie relevantných knižníc v programovacom jazyku Python vyžadovala neúmernú časovú investíciu. Vzhľadom na jednoznačne negatívne výsledky základnej heuristiky je navyše pravdepodobné, že ani aplikácia týchto pokročilejších metód by nepriniesla signifikantne lepšie výsledky.

Hranica istoty	**Počet iterácií**	**F1 skóre train**	**F1 skóre val.**	**Brierovo skóre train**	**Brierovo skóre val.**	**Počet pridaných dát**
Normálny XGBoost		0.6951	0.6408	0.4513	0.5029	
75%	10	0.6309	0.5831	0.5144	0.5827	6 000
80%	10	0.6463	0.6079	0.4847	0.5567	4 527
85%	10	0.6959	0.6235	0.4455	0.5061	212
87.5%	10	0.6962	0.6300	0.4483	0.5035	65
90%	10	0.6967	0.6320	0.4498	0.5047	14

Tabuľka 4: Vplyv prahovej hodnoty istoty na výkonnosť modelu pri metóde samoučenia (10 iterácií) a porovnanie so základným modelom XGBoostu

5.2 Šírenie označení (Label Propagation)

Napriek tomu, že metóda samoučenia nepriniesla signifikantné zlepšenie, ako ďalší prístup sme aplikovali metódu šírenia označení. Pri teoretickom uchopení tohto algo-

ritmu sme vychádzali zo zdrojov [19] a [5]. Tento algoritmus je založený na konštrukcii grafu, v ktorom uzly reprezentujú jednotlivé označené a neoznačené pozorovania (X_L a X_U) a hrany vyjadrujú mieru ich vzájomnej podobnosti. Informácia o triede sa následne iteratívne šíri z označených uzlov pozdĺž hrán do zvyšku grafu, čím dochádza k priradeniu tried neoznačeným pozorovaniam. Základným predpokladom tejto metódy je, že dáta s vysokou mierou vzájomnej podobnosti (blízke uzly) by mali s najvyššou pravdepodobnosťou patriť do rovnakej triedy. Práve z tohto konceptu vychádza pomenovanie šírenie označení.

Váha hrany w_{ij}, predstavujúca podobnosť medzi uzlami i a j, je štandardne definovaná prostredníctvom:

$$w_{ij} = e^{-\frac{\sum_{p=1}^{P}\left(x_{ip}-x_{jp}\right)^2}{2\sigma^2}}$$

Alternatívne môže byť podobnosť definovaná na základe algoritmu K-najbližších susedov (KNN), kde $w_{ij} = 1$, ak je x_i medzi K najbližšími susedmi pozorovania x_j, a $w_{ij} = 0$ inak.

Pomocou týchto podobností sa konštruuje matica pravdepodobností prechodu $T \in \mathbb{R}^{(N+S)\times(N+S)}$, kde prvok $T_{ij} = \frac{w_{ij}}{\sum_{l=1}^{N+S} w_{lj}}$ predstavuje pravdepodobnosť prechodu označenia z pozorovania x_j do pozorovania x_i. Zavedieme tiež maticu $Y_{prob} \in \mathbb{R}^{(N+S)\times K}$, ktorej i-ty riadok reprezentuje pravdepodobnostnú distribúciu príslušnosti pozorovania x_i k jednotlivým triedam.

Algoritmus následne postupuje iteračne: pre počiatočný odhad Y^0_{prob} sa označenia propagujú pomocou matice prechodu ako $Y^1_{prob} = TY^0_{prob}$. Pravdepodobnosti v matici Y^1_{prob} sú následne normalizované po riadkoch tak, aby ich súčet v každom riadku bol rovný 1. Kľúčovým krokom je, že v prípade prvých N riadkov, ktoré prislúchajú pôvodným označeným dátam, sa po každej iterácii obnoví ich pôvodné označenie. Tento krok vychádza z predpokladu, že pôvodné označenia sú exaktné a nemenné. Ak teda napríklad platí $y_i = 3$, po iterácii sa príslušný riadok vráti do stavu $Y^1_{prob,i} = (0, 0, 1, 0, \ldots, 0)$. Tieto kroky sa opakujú až do momentu konvergencie algoritmu.

V [19] bolo matematicky dokázané, že tento iteračný proces konverguje k analytickému riešeniu. V n-tom kroku sa vykoná propagácia $Y^{n+1}_{prob} = \overline{T}Y^n_{prob}$, kde $\overline{T}$ je riadkovo normalizovaná matica prechodu, teda $\overline{T}_{ij} = \frac{T_{ij}}{\sum_{l=1}^{N+S} T_{il}}$. Keďže hodnoty označe-

ných pozorovaní zostávajú konštantné, proces propagácie je možné maticovo prepísať do blokového tvaru:

$$\begin{bmatrix} Y_{prob,L} \\ Y^{n+1}_{prob,U} \end{bmatrix} = \begin{bmatrix} \overline{T}_{LL} & \overline{T}_{LU} \\ \overline{T}_{UL} & \overline{T}_{UU} \end{bmatrix} \begin{bmatrix} Y_{prob,L} \\ Y^{n}_{prob,U} \end{bmatrix}$$

Z tohto systému rovníc je pre účely predikcie relevantná iba časť prislúchajúca neoznačeným dátam:

$$Y^{n+1}_{prob,U} = \overline{T}_{UL} Y^{n}_{prob,L} + \overline{T}_{UU} Y^{n}_{prob,U}$$

Aplikovaním limity pre $n \to \infty$ rovnica prejde do tvaru:

$$Y_{prob,U} = \lim_{n\to\infty} \overline{T}^{n}_{UU} Y^{0}_{prob,U} + \left[\sum_{i=1}^{n} \overline{T}^{i-1}_{UU}\right] \overline{T}_{UL} Y_{prob,L}$$

V citovanej literatúre je dokázané, že $\lim_{n\to\infty} \overline{T}^{n}_{UU} Y^{0}_{prob,U} = 0$. Využitím tohto poznatku a vzorca pre súčet nekonečného geometrického radu matíc získame finálne exaktné riešenie:

$$Y_{prob,U} = (I - \overline{T}_{UU})^{-1} \overline{T}_{UL} Y_{prob,L}$$

Tento algoritmus bol následne aplikovaný na analyzovaný dataset. Ako váhy hrán sme použili 10 najbližších susedov. Po ukončení matematickej propagácie označení boli do množiny označených dát vyextrahované a pridané výlučne tie neoznačené pozorovania, pri ktorých predikovaná pravdepodobnosť (istota) dosiahla prahovú hodnotu minimálne 80%. Na takto obohatenej množine dát boli opätovne, pomocou 5-násobnej krosvalidácie, vyhľadané optimálne parametre a následne natrénovaný finálny model XGBoost.

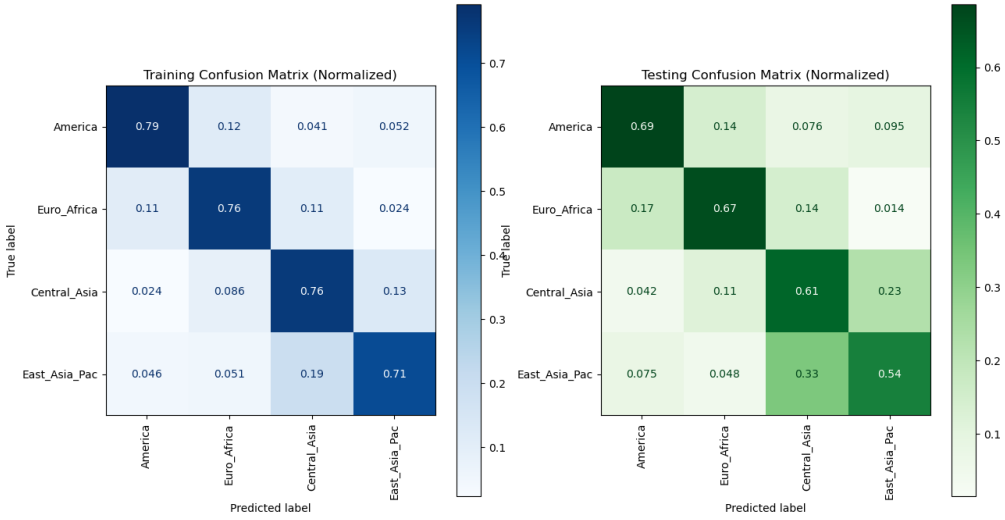
Z klasifikačného reportu a matice zámien (Obr. 37) vyplýva, že hoci model na trénovacej sade vykazuje zlepšenie presnosti v porovnaní so základným modelom XGBoost, na testovacej sade takéto zlepšenie nie je badateľné. Z týchto výsledkov možno usudzovať, že model vykazuje známky mierneho pretrénovania (overfittingu). Tento predpoklad potvrdzuje aj kalibračný graf podľa regiónov (Obr. 38). Zatiaľ čo celková istota modelu narástla (posun hodnôt na osi x doprava), skutočná priemerná presnosť na neznámych testovacích dátach sa nezlepšila. Kombinácia šírenia označení a modelu XGBoost v tomto prípade viedla k vyššej miere „prehnanej sebadôvery“ klasifikátora na úkor jeho reálnej generalizačnej schopnosti.

Classification Report (Train)

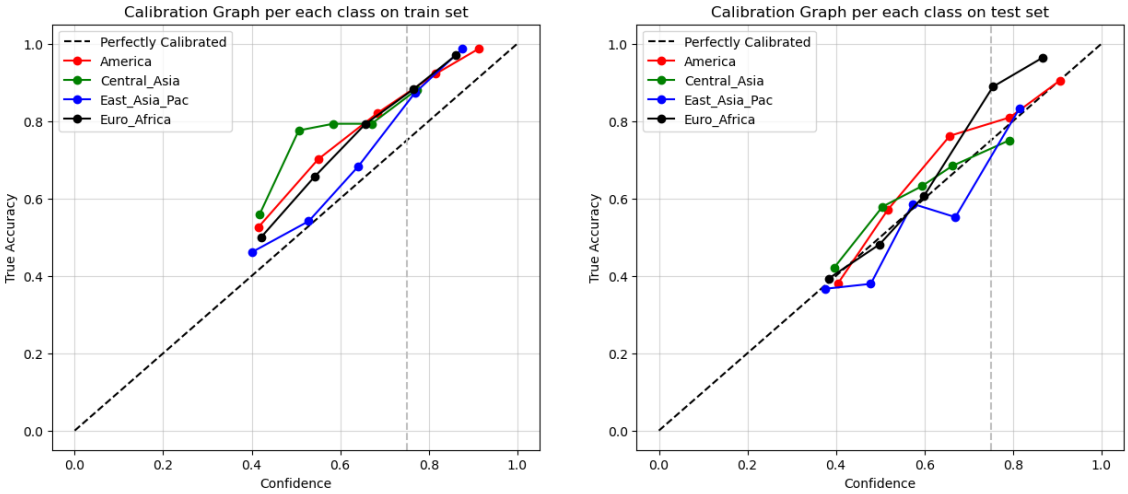
	precision	recall	f1-score	support
America	0.76	0.79	0.77	388
Euro_Africa	0.50	0.76	0.60	292
Central_Asia	0.89	0.71	0.79	788
East_Asia_Pac	0.78	0.76	0.77	509
accuracy			0.75	1977
macro avg	0.73	0.76	0.73	1977
weighted avg	0.78	0.75	0.75	1977

Classification Report (Test)

	precision	recall	f1-score	support
America	0.65	0.69	0.67	105
Euro_Africa	0.43	0.61	0.51	96
Central_Asia	0.70	0.54	0.61	147
East_Asia_Pac	0.74	0.67	0.70	138
accuracy			0.62	486
macro avg	0.63	0.63	0.62	486
weighted avg	0.65	0.62	0.63	486



Obr. 37: Klasifikačný report a matica zámien modelu XGBoost pre trénovaciu množinu, obsahujúcu pôvodné označené dáta a pridané neoznačené po algoritme Label Propagation, a testovaciu množinu



Obr. 38: Kalibračný graf modelu XGBoost rozdelený podľa cieľových regiónov

Záver

Hlavným cieľom tejto práce bol návrh a implementácia algoritmu schopného predikovať geografickú polohu, respektíve časové pásmo, jednotlivých používateľov siete Bitcoin. Tento cieľ bol úspešne naplnený v štvrtej kapitole. Pre natrénovanie takéhoto klasifikačného modelu bolo kľúčové vytvoriť robustnú databázu s exaktne priradenými časovými pásmami. Vzhľadom na inherentnú pseudonymitu siete Bitcoin predstavoval tento krok významnú metodologickú výzvu.

Spočiatku sme zvažovali využitie dát z búrz uplatňujúcich reštrikcie pri registrácii používateľov. Nakoniec sme sa pri zbere dát inšpirovali metodikou z článku [11]. Hoci primárnym zameraním spomenutého článku nebola geografická lokalizácia používateľov, jej aplikačná časť demonštrovala extrakciu dát z kryptomenového diskusného fóra Bitcointalk. Po analýze profilov sme zistili, že časť používateľov okrem bitcoinovej adresy dobrovoľne uvádza aj svoju reálnu polohu. Keďže na platforme je registrovaných viac ako 3,7 milióna členov, poskytla nám táto metóda (popísaná v Kapitole 2) dostatočne veľkú vzorku adries s označením polohy.

Zvýšený dopyt po službách centralizovaných búrz pri prvotnom získavaní bitcoinov spôsobil, že značná časť extrahovaných adries patrí práve týmto inštitúciám. Problematika búrz spočíva v tom, že hoci generujú unikátne adresy pre jednotlivých klientov, ich celková správa zostáva centralizovaná. Transakčná aktivita na týchto adresách preto nemusí nutne odzrkadľovať reálne vykonávanie transakcií používateľa. Z tohto dôvodu bolo nevyhnutné takéto adresy z datasetu vylúčiť. Tento krok sme realizovali pomocou komerčnej služby Arkham Intelligence, ktorá sa špecializuje na deanonymizáciu a označovanie adries verejných subjektov (burzy, mixovacie služby a iné komerčné platformy). Po odfiltrovaní búrz sme dataset obohatili aplikovaním metód heuristického zhlukovania predstavených v [11], čím sme k jednotlivým používateľom priradili ďalšie adresy patriace do ich kontroly a získali tak rozsiahlejšiu transakčnú históriu.

Následne prebehol zber samotných transakcií ku všetkým identifikovaným adresám priamo z prostredia blockchainu. Z technických dôvodov a vysokej hardvérovej náročnosti sme neprevádzkovali vlastný plnohodnotný uzol. Namiesto toho sme využili API rozhranie blockchainového prehliadača, ktoré nám sprostredkoval kom-

pletnú transakčnú históriu. Agregáciou týchto údajov vznikla finálna databáza, v ktorej sme aktivitu každého používateľa popísali pomocou 24 premenných, reprezentujúcich relatívny podiel vykonaných transakcií v jednotlivých hodinách dňa.

Analýzou týchto dát sme potvrdili predpoklad, že kľúčovým diskriminačným znakom pre určenie časového pásma je existencia a časové umiestnenie okna neaktivity (spánku). Zároveň sme očakávali, že šírka tohto okna a jeho prekrývanie znemožní modelom spoľahlivo rozlišovať používateľov z úzko susediacich časových pásiem. Z tohto dôvodu, a tiež pre nerovnomerné demografické rozloženie vzorky, sme pôvodné časové pásma agregovali do štyroch makro-regiónov: Amerika, Euroafrika, Centrálna Ázia a Východná Ázia s Pacifikom.

Na takto pripravené dáta boli následne aplikované modely logistickej regresie a algoritmus XGBoost. Výber týchto dvoch klasifikačných prístupov bol podmienený potrebou predikovať nielen samotnú triedu, ale aj pravdepodobnosť (istotu) zaradenia do tejto triedy. Pri analýze výsledkov sme sa zamerali primárne na predikcie s najvyššou mierou istoty modelu. Logistická regresia, plniaca funkciu referenčného modelu, vykazovala ťažkosti pri rozlišovaní Centrálnej a Východnej Ázie. Nasadením robustnejšieho algoritmu XGBoost a hľadaním jeho optimálnych hyperparametrov sme sa pokúsili túto chybovosť znížiť.

Model XGBoost priniesol mierne zlepšenie výkonnosti. Na základe manuálnej analýzy predikcií na testovacej sade sme dospeli k záveru, že zlyhania modelu pri vysokých istotách boli primárne spôsobené anomálnym biorytmom konkrétnych jednotlivcov, ktorí sa vymykali štandardnému cyklu svojho regiónu. Pri nižších hladinách istoty však aj tento pokročilý model narazil na identický dátový limit ako logistická regresia – zníženú schopnosť separovať Centrálnu a Východnú Áziu. Príčinou je úzka fázová blízkosť časových pásiem, ktoré tieto regióny formujú. Táto skutočnosť spätne validuje náš prvotný metodický krok agregácie dát. Potvrdzuje sa tým hypotéza, že na základe časovej distribúcie transakcií nie je možné dosiahnuť mimoriadne vysokú granularitu lokalizácie. Rozšírenie modelu na viacero úzkych časových pásiem by viedlo k poklesu presnosti. Napriek týmto objektívnym limitom však navrhnutý klasifikačný model hodnotíme ako funkčný a vo všeobecnosti dostatočne presný v kontexte poskytnutej dátovej sady.

V záverečnej fáze sme otestovali využitie metód Semi-supervised learningu, ktoré kombinujú označené a neoznačené pozorovania. Tento prístup mal vernejšie simulovať reálnu štruktúru blockchainu, kde potvrdenou lokalitou disponujeme iba pri zlomku dostupných dát. Ako neoznačenú vzorku sme využili profily používateľov z diskusného fóra, ktorí uviedli svoju bitcoinovú adresu, no chýbala u nich geografická informácia. Implementácia týchto metód však nepriniesla očakávané zlepšenie výkonnosti. Naopak, klasifikačná presnosť pôvodného modelu sa mierne zhoršila.

Zoznam použitej literatúry

[1] Akiba, T., Sano, S., Yanese, T., Ohta, T., Koyama, M.: *Optuna: A Next-generation Hyperparameter Optimization Framework*, Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2019), 2623-2631

[2] Antonopoulos, A. M.: *Mastering Bitcoin*, O'Reily Media, California, 2017

[3] Badev, A., Chen, M.: *Bitcoin: Technical Background and Data Analysis*, FEDS working paper No. 2014-104, 2014, dostupné na internete (3.1.2026): https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2544331

[4] Blockchainový prehliadač poskytujúci prístup k celkovej transakčnej histórií adries, dostupné na internete (6.5.2026): https://www.blockchain.com/explorer

[5] Chapelle, O., Schölkopf, B., Zien, A.: *Semi-supervised learning*, MIT Press, Cambridge, Massachusetts, London, England, 2006

[6] Chen, T., Guestrin, C.: *Xgboost: A scalable tree boosting system*, Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining (2016), 785-794

[7] Diskusné fórum Bitcointalk primárne určené pre kryptomenu Bitcoin, dostupné na internete (6.5.2026): https://bitcointalk.org/

[8] Friedman, J.: *"Greedy Function Approximation: A Gradient Boosting Machine*, The Annals of Statistics, vol. 29, no. 5 (2001), 1189-1232

[9] Hamala, M., Trnovská, M.: *Nelineárne programovanie, teória a algoritmy*, Epos, Bratislava, 2013

[10] Hastie, T., Tibshirani, R., Friedman, J.: *The Elements of Statistical Learning*, Springer series in statistics, New York, 2009

[11] He, X., He, K., Lin, S., Yang, J., Mao, H.: *Bitcoin address clustering method based on multiple heuristic conditions.*, IET Blockchain 2 (2022), 44-56

[12] James, G., Witten, D., Hastie, T., Tibshirani, R., Taylor, J.: *An introduction to Statistical Learning: With applications in Python*, Springer, 2023

[13] Learn Me A Bitcoin, dostupné na internete (7.1.2026): https://learnmeabitcoin.com/

[14] Nakamoto, S.: *Bitcoin: A Peer-to-Peer Electronic Cash System*, Working Paper, 2008, dostupné na internete (3.1.2026): https://bitcoin.org/en/bitcoin-paper

[15] Narayanan, A., Bonneau, J., Felten, E., Miller, A., Goldfeder, S.: *Bitcoin and Cryptocurrency Technologies*, Princeton University Press, New Jersey, 2020

[16] Používateľská príručka ku knižnici scikit learn v pythone, dostupné na internete (3.4.2026): https://scikit-learn.org/stable/user_guide.html

[17] Triguero, I., García, S., Herrera, F.: *Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study*, Knowledge and Information systems 42.2 (2015), 245-284

[18] Zhu, X.: *Semi-Supervised Learning Literature Survey*, Technical report, Computer Sciences, University of Wisconsin-Madisoa, 2008, dostupné na internete (24.4.2026): https://pages.cs.wisc.edu/~jerryzhu/pub/ssl_survey.pdf

[19] Zhu, Z., Zoubin, G.: *Learning from labeled and unlabeled data with label propagation*, Technical Report CMU-CALD-02–107. Carnegie Mellon University, 2002, dostupné na internete (25.4.2026): https://pages.cs.wisc.edu/~jerryzhu/pub/CMU-CALD-02-107.pdf

[20] Zoznam Bitcoin podujatí na SLovensku, dostupné na internete (6.5.2026): https://www.dvadsatjeden.org/kalendar/

Databáza a kódy

Kompletnú databázu bitcoinových transakcií, ktorých zber sme vysvetlili v Kapitole 2, sme uchovávali na lokálnom MySQL serveri. Z technických dôvodov nie je možné poskytnúť k tomuto serveru verejný prístup. Celú databázu sme preto sprístupnili vo forme exportu vo formáte .sql na platforme Kaggle:https://www.kaggle.com/datasets/matejdukat/bitcoin-a-time-zone-analysis/data. Databáza pozostáva zo šiestich tabuliek:

- **scraped_data:** Obsahuje informácie o profiloch z diskusného fóra Bitcointalk, ako je verejná BTC adresa, uvádzaná geografická poloha, počet transakcií, dátum poslednej transakcie a časová zóna (3 700 001 riadkov).
- **exchanges_detection:** Uchováva informácie o priradení konkrétnych adries k burzám alebo iným komerčným kryptomenovým platformám, ktoré boli získané na základe služby Arkham Intelligence (55 237 riadkov).
- **entities_heur:** Predstavuje výsledky heuristického zhlukovania. V tejto tabuľke sú jednotlivé bitcoinové adresy prepojené a zoskupené pod spoločné entity na základe ich identifikačných čísel (443 498 riadkov).
- **blockchain_data:** Poskytuje záznamy o všetkých transakciách patriacim entitám z tabuľky entities_heur (viac detailov v časti 2.2) (14 172 858 riadkov)
- **tx_inputs:** Obsahuje detailné informácie o všetkých transakčných vstupoch k príslušným záznamom z tabuľky blockchain_data (viac detailov v časti 2.2) (81 514 681 riadkov)
- **tx_outputs:** Obsahuje detailné informácie o všetkých transakčných výstupoch (novovytvorených UTXO) k príslušným záznamom z tabuľky blockchain_data (viac detailov v časti 2.2) (157 701 162 riadkov)

Okrem surových dát sme na platforme zverejnili aj spracované datasety vo formáte .csv, ktoré slúžia ako priamy vstup pre klasifikačné modely. Tieto súbory umožňujú rýchlu manipuláciu a replikáciu výsledkov bez nutnosti inštalácie a lokálneho spúšťania MySQL servera. Ide o dva súbory:

- **database_from_Bitcointalk.csv:** Dataset s finálnymi dátami využitý pri trénovaní logistickej regresie a XGBoostu. Okrem 24 stĺpcov, ktoré reprezentujú relatívne počty transakcií v jednotlivých hodinách dňa, a stĺpca s priradeným cieľovým regiónom, obsahuje aj celkový počet transakcií a pomocné premenné slúžiace na detekciu neaktívneho časového okna používateľa. Tieto pomocné premenné neboli priamo použité vo fázach trénovania klasifikátorov.

- **database_from_Bitcointalk_unlabeled.csv:** Dataset s totožnou vnútornou štruktúrou ako predchádzajúci súbor, avšak bez priradených cieľových tried (regiónov). Reprezentuje entity, pri ktorých nebolo možné jednoznačne určiť reálne časové pásmo. Tento celok dát bol primárne využitý pri aplikácii semi-supervised learningu.

Na prácu s databázou, transformáciu dát a samotnú implementáciu modelov strojového učenia bol použitý programovací jazyk Python. Všetky zdrojové kódy a skripty (Jupyter Notebooky) sú verejne dostupné v našom repozitári na platforme GitHub: https://github.com/MatejDuk/analyza_casovych_pasiem/tree/main. Úložisko obsahuje nasledujúce súbory:

- **web_scraping_btc_addresses_from_bitcointalk_forum:** Skript zabezpečujúci automatizovaný zber dát z profilov používateľov na fóre Bitcointalk, ktoré sú následne ukladané do tabuľky scraped_data. Pre správne fungovanie je potrebné pripojenie k MySQL serveru a implementáciu rotácie proxy serverov (IP adries).

- **blockchain_data_to_forum_addresses:** Kód slúžiaci na hromadné sťahovanie transakčnej histórie ku všetkým identifikovaným bitcoinovým adresám prostredníctvom API blockchainového prehliadača. Podobne ako pri predchádzajúcom skripte, aj tento proces je podmienený prístupom k databáze a využívaním proxy serverov.

- **heuristic_clustering:** Algoritmus realizujúci heuristické zhlukovanie, ktorý identifikuje prepojenia medzi vstupnými adresami z fóra a následne ich agreguje do ucelených entít. Výpočtový proces prebieha priamo nad dátami zo spusteného MySQL servera.

- **database_after_heuristic_clustering:** Kód, ktorý zo surových dát z MySQL serveru vytvorí dáta pripravené na aplikovanie klasifikačných modelov. Pre správne fungovanie je potrebný prístup k MySQL serveru.

- **classification_models:** Kód, kde sú trénované a následne predstavené výsledky logistickej regresie a XGBoostu ako aj hľadanie optimálnych parametrov. Pre spustenie tohto kódu je postačujúce stiahnutie súboru: database_from_Bitcointalk.csv.

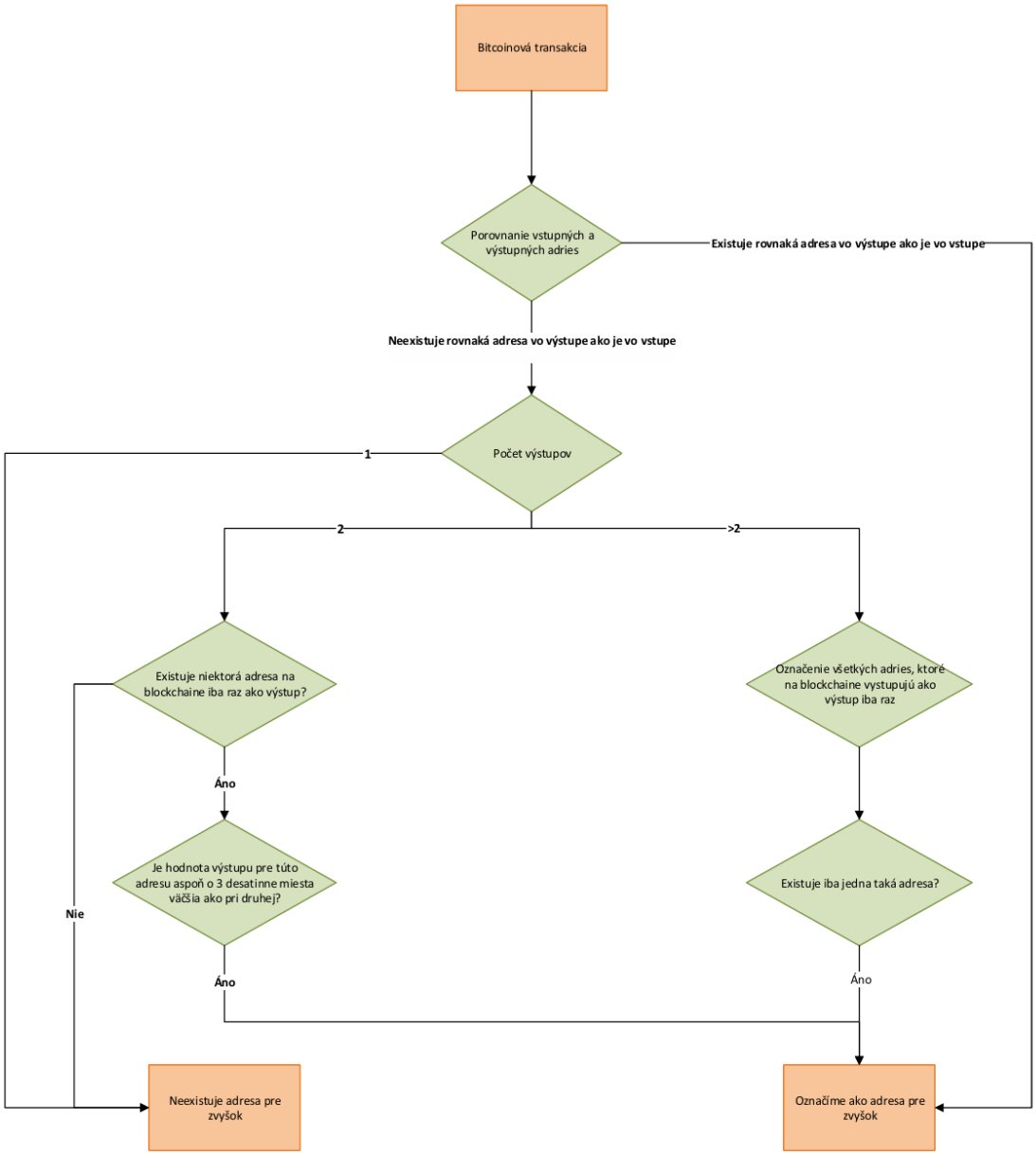
- **semi_supervised_learning:** Kód, kde sú trénované a následne predstavnené výsledky semi supervised learninug (Self learninig a Label Propagation). Pre spustenie tohto kódu je postačujúce stiahnutie súborov: database_from_Bitcointalk.csv a database_from_Bitcointalk_unlabeled.csv

Príloha A


```
"hash160": "376b1b055fc388c6a10c8a4f0cd9dbf529e62313",
"address": "bc1qxa43kp2lcwyvdggv3f8sekwm7557vgcnqle25r",   #adresa, ktorú skúmame a zadávame, za $bitcoin_address
"n_tx": 2,                #počet transakcií
"n_unredeemed": 0,
"total_received": 16800,  #celková výška prijatých BTC vyjadrených v Satoshi, 1BTC = 100 000 000 Satoshi
"total_sent": 16800,      #celková výška odoslaných BTC vyjadrená v Satoshi
"final_balance": 0,       #výsledný zostatok na adrese v Satoshi
"txs": [            #doteraz výstup obsahoval základné informácie o adrese, teraz nasleduje časť z jednotlivými transakciami
  {
    "hash": "c02e7d778f8aeccb6594409389345e3679c8d499f5d6d4005b834af68ee91055",  #txid
    "ver": 2,
    "vin_sz": 1,                                                                 #počet vstupov
    "vout_sz": 2,                                                                #počet výstupov
    "size": 222,                                                                 #veľkosť transakcie v bytoch
    "weight": 561,                                                               #weight/4 = virtual bytes
    "fee": 200,                                                                  #poplatok za transakciu
    "relayed_by": "0.0.0.0",
    "lock_time": 928893,
    "tx_index": 2992996041541968,
    "double_spend": false,
    "time": 1766354299,  #čas v tomto prípade vyjadrujúci kedy prišla transakcia do memory poolu, do bloku 725172 to bol čas schválenia bloku
    "block_index": 928894,
    "block_height": 928894,                                                      #poradové číslo bloku, v ktorom bola transakcia schválená
    "inputs": [     #nasledujúca časť obsahuje dáta o vstupoch do transakcie
      {
        "sequence": 4294967293,
        "witness": "02473044022049bd721064da3ed1982bc13aaced76ad3d67812dc18c8ac7128394c5e397f824022009a0f8c542bf647a42199cdc2c407236de85460ddd
        "script": "",
        "index": 0,
        "prev_out": {
          "type": 0,
          "spent": true,
          "value": 16800,                                                        #hodnota vstupu, nevyčerpaného UTXO v Satoshi
          "spending_outpoints": [···
          "n": 1,
          "tx_index": 5532586128763964,
          "script": "0014376b1b055fc388c6a10c8a4f0cd9dbf529e62313",
          "addr": "bc1qxa43kp2lcwyvdggv3f8sekwm7557vgcnqle25r"                    #adresa, z ktorej sa čerpajú UTXO
        }
      }
    ],
    "out": [      #nasledujúca časť obsahuje dáta o výstupoch transakcie
      {
        "type": 0,
        "spent": false,
        "value": 2000,                                                           #hodnota výstupu, nového UTXO v Satoshi
        "spending_outpoints": [],
        "n": 0,
        "tx_index": 2992996041541968,
        "script": "0014f27a21ba92520510d82da6d629534a6710b29c5a",
        "addr": "bc1q7fazrw5j2gz3pkpd5mtzj562vugt98z6h20xn7"                      #adresa, na ktorú sa odosielajú BTC
      },
      {···
    ],
    "result": -16800,
    "balance": 0
  },
```


Obr. 39: Opísaný príklad výstupu z Blockchain API

Príloha B



Obr. 40: Vývojový diagram metódy heuristického zhlukovania H2